Learning Debiased Representation via Disentangled Feature Augmentation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Image classification models tend to make decisions based on peripheral attributes of data items that have strong correlation with a target variable (i.e., *dataset bias*). These *biased* models suffer from the poor generalization capability when evaluated on unbiased datasets. Existing approaches for debiasing often identify and emphasize those samples with no such correlation (i.e., *bias-conflicting*) without defining the bias type in advance. However, such bias-conflicting samples are significantly scarce in biased datasets, limiting the debiasing capability of these approaches. This paper first presents an empirical analysis revealing that training with "diverse" bias-conflicting samples beyond a given training set is crucial for debiasing as well as the generalization capability. Based on this observation, we propose a novel feature-level data augmentation technique in order to synthesize diverse bias-conflicting samples. To this end, our method learns the disentangled representation of (1) the *intrinsic attributes* (i.e., those inherently defining a certain class) and (2) *bias attributes* (i.e., peripheral attributes causing the bias), from a large number of *bias-aligned* samples, the bias attributes of which have strong correlation with the target variable. Using the disentangled representation, we synthesize bias-conflicting samples that contain the diverse intrinsic attributes of bias-aligned samples by swapping their latent features. By utilizing these diversified bias-conflicting features during the training, our approach achieves superior classification accuracy and debiasing results against the existing baselines on both synthetic as well as real-world datasets.

1 Introduction

Despite the recent advancement of deep neural networks, they often rely overly on the correlation between peripheral attributes and labels, referred to as *dataset bias* [1], especially when such strong bias is found in a given dataset. A majority of samples in the biased dataset exhibit visual attributes that are not innate but frequently co-occur with target labels (i.e., *bias attributes*). For example, most of the bird images in the training dataset may contain the background as the blue sky, while the birds may still be found in different places. Thus, the model trained with such a biased dataset is likely to learn the bias attributes more than *intrinsic attributes*, the innate visual attributes that inherently define a certain class, e.g., the wings of birds. This causes the model to learn shortcuts for classification [2], failing to generalize on minor images with no such correlations (e.g., birds on grounds or grass) during the test phase. Throughout the paper, *bias-aligned* samples correspond to those containing a strong correlation between bias attributes and labels (e.g., birds on the sky), while *bias-conflicting* samples indicate the other cases that are rarely found (e.g., birds on grounds)

To tackle such a task, previous studies often define a specific bias type (e.g., color and texture) in advance [3, 4, 5, 6, 7, 8, 9, 10], which enables them to design a debiasing network tailored for the

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

bias. For example, Bahng *et al.* [6] leverage BagNet [11], which has limited size of receptive fields, to focus on learning color and texture. However, defining a bias type in advance 1) limits the capability of debiasing in other bias types and 2) requires expensive labor to manually identify the bias type. To handle such an issue, a recent approach [12] defines a bias based on an intuitive observation that the bias attributes are often *easier* to learn than the intrinsic attributes for neural networks. In this regard, they re-weight bias-conflicting samples while de-emphasizing the bias-aligned ones. However, we point out that the reason behind the limited generalization capability of existing debiasing approaches lies in the significant scarcity of bias-conflicting samples compared to the bias-aligned ones in a given training set. In other words, it is challenging to learn the debiased representation from these scarce conflicting samples because the models are prone to memorize (thus being overfitted to) these samples, failing to learn the intrinsic attributes. Therefore, we claim that a neural network can learn properly debiased representation when these data items are diversified during training.

We conduct a brief experiment to demonstrate the importance of *diversity* in debiasing. Diversity in our work indicates different valid realization of intrinsic attributes in a certain class e.g., different thickness or tilted degrees of digits in MNIST [13]. Our observation is that training a model with diverse bias-conflicting samples beyond a given training set is crucial for learning debiased representation (Section 3.2). In this regard, synthesizing bias-conflicting samples is one of the straightforward approaches to increase the diversity of such samples. In fact, a large amount of bias-aligned samples in a given training set already contain diverse intrinsic attributes, which can work as informative sources for increasing the diversity. However, as bias and intrinsic attributes are highly entangled in their embedding space, it is difficult to extract the intrinsic ones from these bias-aligned samples. Therefore, disentangling these correlations enables to synthesize diversified bias-conflicting samples that originate from bias-aligned samples.

In this paper, we propose a novel feature augmentation approach via disentangled representation for debiasing. We first train two different encoders to embed images into the disentangled representation of their intrinsic and bias attributes. With the disentangled representation, we randomly swap the latent vectors extracted from different images, most of which are bias-aligned samples in our training set. These swapped features thus contain both bias and intrinsic attributes without the correlation between them, which, in turn, can work as augmented bias-conflicting samples in our training. These features include intrinsic features of bias-aligned ones, increasing the diversity that a given training set lacks. Therefore, these augmented samples include intrinsic attributes which cannot be obtained from the existing bias-conflicting samples, increasing the diversity. Furthermore, to enhance the quality of diversified features, we schedule the feature augmentation after the representation is disentangled at a certain degree. The main contributions of our work include:

- Through a simple experiment, we find that the diversity of bias-conflicting samples is crucial for debiasing.
- Based on such an observation, we propose a novel feature augmentation method via disentangled representation for diversifying the bias-conflicting samples.
- We achieve the state-of-the-art performances in two synthetic datasets (i.e., Colored MNIST and Corrupted CIFAR10) and two real-world datasets (i.e., Biased FFHQ and BAR) against existing baselines.

2 Related Work

Debiasing predefined bias Several existing approaches mitigate the bias by pre-defining a certain bias type, either explicitly [3, 4, 5] or implicitly [6, 7, 8, 9, 10, 14]. For example, Bahng *et al.* [6] and Wang *et al.* [7] design a color and texture-oriented network to adversarially learn a debiased model against the biased one. However, as these methods still require a specific bias type such as texture, they lack the general applicability to the datasets where the bias types are demanding to recognize.

Instead of defining certain types of bias, recent approaches [12, 15, 16] rely on the straightforward assumption that networks are prone to exploit the bias when it acts as a shortcut [2], i.e. easy to learn in the early training phase. Nam *et al.* [12] emphasize the bias-conflicting samples during training by using generalized cross-entropy loss [17]. Darlow *et al.* [15] and Huang *et al.* [16] presume that high gradient of latent vectors accounts for the shortcuts that model learns. In the line with the recent studies, we tackle debiasing without pre-defining a certain bias type.

Dataset	Diversity ratio	Sampling ratio	Accuracy.
Colored-MNIST	5%	50%	**83.77**$\pm$2.03
	1%	50%	67.19$\pm$1.99
	5%	1%	<u>77.97</u>$\pm$6.00
Corrupted CIFAR10	5%	50%	**46.99**$\pm$0.82
	1%	50%	33.08$\pm$0.80
	5%	1%	<u>36.66</u>$\pm$0.55

Table 1: The classification accuracy on the unbiased test sets. The diversity ratio indicates the ratio of bias-conflicting samples in the dataset pooled for each experiment. Sampling ratio refers to the ratio of bias-conflicting samples included in each mini-batch. We report the averaged accuracy over three independent trials with the standard deviation. In both datasets, we observe that the bias can be mitigated with diverse bias-conflicting samples even with a small sampling ratio. Bold and underlined values indicate the best and second best accuracy, respectively.

Data augmentation for debiasing Geirhos *et al.* [10] mitigates the texture bias by utilizing additional training images with their styles being transferred by adaptive instance normalization (AdaIN) [18]. Minderer *et al.* [19] trains an image-to-image translation network for removing shortcut cues in the self-supervised task. However, such image-level data augmentation is limited to resolving the predefined texture bias which can not be adopted to other general types of bias.

One alternative is to exploit the latent space for data augmentation. For example, Darlow *et al.* [15] adversarially perturb the latent vectors corresponding to the high gradients to generate the samples against bias. Zhou *et al.* [20] mixes the style of different source domains by AdaIN [18] to increase the domain generalization ability. Albeit the effectiveness of the augmentation in the latent space, the strong unwanted correlation between bias attributes and labels prevents from obtaining the desirable intrinsic features. We resolve this issue by leveraging the disentangled representation in debiasing, which is widely used in image-to-image translation task [21, 22, 23]. To the best of our knowledge, no previous work in debiasing leverage this disentangled representation for the purpose of feature augmentation. For the rest of the paper, we elaborate how we perform the feature augmentation based on the disentangled representation.

3 Importance of Diversity in Debiasing

This section describes the details of a toy-set experiment in which we observe the importance of diversity in learning debiased representation. We first introduce the two synthetic datasets, Colored MNIST and Corrupted CIFAR10, that we utilize for the observation in Section 3.1. Then we elaborate the results of the experiments in Section 3.2.

3.1 Dataset

Colored MNIST is a modified MNIST dataset [13][1] with the color bias. We select ten distinct colors and inject each color on the foreground of each digit to create color bias. By adjusting the number of bias-conflicting data samples among the training set, we obtain four different datasets with the ratio of bias-conflicting samples of 0.5%, 1%, 2%, and 5%.

Corrupted CIFAR10[2] has ten different types of texture bias applied in CIFAR10 [24] dataset, constructed by following the design protocol of Hendrycks and Dietterich [25]. Each class is highly correlated with a certain texture (e.g., frost and brightness). Corrupted CIFAR10 also has four different datasets with their correlation ratios as in Colored MNIST.

3.2 Increasing diversity outperforms oversampling

To confirm the significance of diversity of bias-conflicting samples in debiasing, we train three different settings: oversampling bias-conflicting samples by 50% in each mini-batch (i.e. 128 from a

[1] CC-BY-SA 3.0
[2] CC-By 4.0

batch size of 256), from the pool of i) 5% dataset, ii) 1% dataset and iii) sampling bias-conflicting samples by 1% in each mini-batch (i.e. 2 from a batch size of 256) from the pool of 5% dataset. Oversampling provides the same amount of bias-conflicting samples as the aligned ones to the model in every training step. Bias-conflicting images sampled from the pool of 5% dataset have more diverse appearances of bias-conflicting samples compared to those from 1% dataset.

Table 1 shows the image classification accuracy of each settings validated on the unbiased test images. Apparently, oversampling diverse bias-conflicting samples (first row) outperforms the other two methods. The interesting finding is that sampling fewer but diverse conflicting samples in each mini-batch (third row) outperforms oversampling bias-conflicting samples with limited diversity (second row). These results lead to the conclusion that the diversity of bias-conflicting samples is a more crucial factor for learning debiased representation than the ratio of sampling in the training. As the diversity is limited (the latter case), the model can be easily overfitted to the given bias-conflicting samples, thus less likely to learn the generalized intrinsic attributes. With Colored MNIST as an example, the model may fail to predict bias-conflicting samples of narrow digits in the test phase when it is trained with bias-conflicting samples mostly composed of thick digits. On the other hand, in the former case, the model can learn multiple visual facets of the bias-conflicting samples (e.g., both thick and narrow digits) when they are sampled from the diverse pool of datasets, resulting in learning intrinsic attributes from them.

4 Debiasing via disentangled feature augmentation

Motivated by such an observation in Section 3.2, we propose a feature-level augmentation strategy for synthesizing additional bias-conflicting samples, as illustrated in Fig. 1. First, we train the two separate encoders which embed an image into disentangled latent vectors corresponding to the intrinsic and bias attributes, respectively (Section 4.1). By swapping these feature vectors among training samples, it enables to augment the bias-conflicting samples which do not have a correlation between two attributes (Section 4.2). To further enhance the effectiveness, we schedule the feature augmentation after the representation is disentangled at a certain degree (Section 4.3).

4.1 Learning disentangled representation

In contrast to the bias-conflicting samples, a large amount of bias-aligned images have diverse appearances of their intrinsic attributes (e.g. tilted digits with diverse degrees). By leveraging these attributes for augmentation, we can naturally obtain the diversified bias-conflicting samples containing the diverse intrinsic attributes. However, it remains challenging in that these attributes are strongly correlated with the bias attributes in the bias-aligned samples. Therefore, we propose to design two encoders with their linear classifiers to extract the disentangled latent vectors from the input images.

As shown in Fig. 1, encoders E_i and E_b embed an image x into intrinsic feature vectors $z_i = E_i(x)$ and bias attributes $z_b = E_b(x)$, respectively. Afterward, linear classifiers C_i and C_b are trained to predict the target labels with the concatenated vector $z = [z_i; z_b]$. To train E_i and C_i as intrinsic feature extractor and E_b and C_b as bias extractor, we utilize the relative difficulty score of each data sample, inspired by the previous work of Nam *et al.* [12]. More specifically, we train E_b and C_b to be overfitted to the bias attributes by utilizing the generalized cross entropy (GCE) [17], while E_i and C_i are trained with the cross entropy (CE) loss. Then, the samples with high CE loss from C_b can be regarded as the bias-conflicting samples compared to the samples with low CE loss. In this regard, we obtain the relative difficulty score of each data sample as

$$W(x) = \frac{CE(C_b(z), y)}{CE(C_i(z), y) + CE(C_b(z), y)}, \tag{1}$$

where y indicate the target labels. As bias-conflicting samples obtain high values of W, we emphasize the loss of these samples for training E_i and C_i, enforcing them to learn intrinsic attributes. Therefore, overall objective functions for disentanglement can be written as

$$L_{\text{dis}} = CE(C_i(z), y) + \lambda_{\text{dis}} GCE(C_b(z), y). \tag{2}$$

To ensure that C_i and C_b predicts target labels mainly based on z_i and z_b, respectively, the loss from

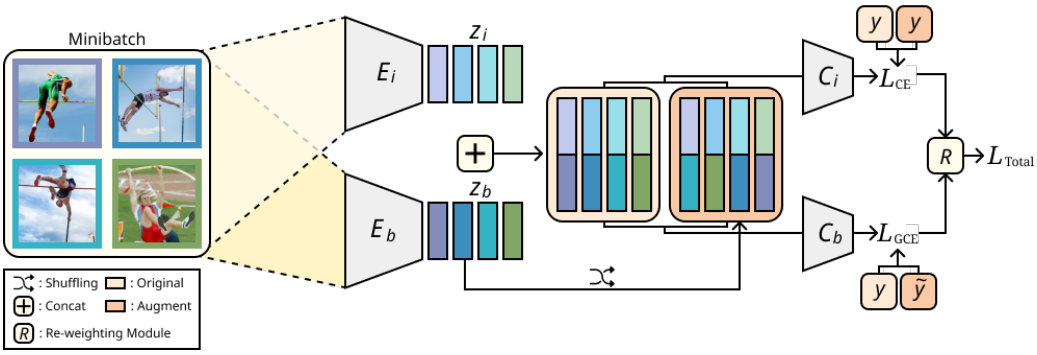

Figure 1: The overview of our proposed debiasing approach. (E_i, C_i) and (E_b, C_b) are pairs of an encoder and a linear classifier trained to learn the disentangled representation of intrinsic attributes and bias attributes, respectively. With the disentangled features z_i and z_b, the feature augmentation is performed by swapping these latent vectors among different training samples, after certain iterations of training. R refers to the re-weighting algorithm which implicitly differentiates bias-aligned samples and bias-conflicting samples. Each color indicates the different data samples.

Algorithm 1 Debiasing with disentangled feature augmentation

Input: image x, label y, iteration t, aug iteration t_{swap}
Initialize two networks (E_i, C_i), (E_b, C_b)
while *not converged* **do**
  Extract z_i, z_b from $E_i(x)$, $E_b(x)$
  Concatenate $z = [z_i; z_b]$
  Update (E_i, C_i), (E_b, C_b) with $L_{\text{dis}} = CE(C_i(z), y) + GCE(C_b(z), y)$
  if $t > t_{\text{swap}}$:
    Randomly shuffle $z = [z_i, z_b]$ into $z_{\text{swap}} = [z_i; \tilde{z}_b]$
    Calculate $L_{\text{swap}} = CE(C_i(z_{\text{swap}}), y) + GCE(C_b(z_{\text{swap}}), \tilde{y})$
    Update (E_i, C_i), (E_b, C_b) with $L_{\text{total}} = L_{\text{dis}} + \lambda_{\text{swap}} L_{\text{swap}}$
end

C_i is not backpropagated to E_b, and vice versa.

4.2 Feature swapping for augmentation

While such an architecture disentangles the intrinsic features and bias features, E_i and C_i are mainly trained with excessively small amounts of bias-conflicting samples. Therefore, E_i and C_i fail to fully acquire the intrinsic representation of a target class. To promote further improvement in learning intrinsic feature vectors, we diversify the bias-conflicting samples by swapping the disentangled latent vectors among the training sets. In other words, we randomly shuffle the intrinsic features and bias features in each mini-batch and obtain $z_{\text{swap}} = [z_i; \tilde{z}_b]$ where $\tilde{z}_b$ denotes the randomly shuffled bias attributes of z_b. As the intrinsic and bias attributes in z_{swap} are obtained from two different images, they certainly have less correlation compared to $z = [z_i; z_b]$ where both are from the same image. Since the biased dataset is mostly composed of bias-aligned samples, these vectors are likely from the bias-aligned samples, highly diversified compared to the bias-conflicting ones. Then, $z_{\text{swap}} = [z_i; \tilde{z}_b]$ act as augmented bias-conflicting latent vectors with diversity inherited from the bias-aligned samples. Along with L_{dis}, we add the following loss function to train two neural networks with the augmented features

$$L_{\text{swap}} = CE(C_i(z_{\text{swap}}), y) + \lambda_{\text{swap}_b} GCE(C_b(z_{\text{swap}}), \tilde{y}) \tag{3}$$

where $\tilde{y}$ denote target labels for shuffled bias attributes $\tilde{z}$ and C_i and C_b are trained to learn intrinsic attributes and bias attributes after random swapping, respectively. Thus, our total loss function is as follows

$$L_{\text{total}} = L_{\text{dis}} + \lambda_{\text{swap}} L_{\text{swap}} \tag{4}$$

where λ_{swap} is adjusted for weighting the importance of the feature augmentation.

Figure 2: Example images of datasets utilized in our work. In each dataset, the first two rows indicate the bias-aligned samples while the third row below the dotted line indicates the bias-conflicting samples.

4.3 Scheduling the feature augmentation

While training with additional synthesized features helps to mitigate the unwanted correlation, utilizing them from the beginning of training does not improve the debiasing performance. To be more specific, in the early stage of training, the representations of z_i and z_b are imperfectly disentangled to be used as the sources of feature augmentation. Feature augmentation should be conducted after two features are disentangled at a certain degree. Without the disentangled representation, the augmented features work as noisy samples which may aggravate the debiasing performances. We verify the importance of scheduling the feature augmentation in Table. 3. Our approach can be summarized with Algorithm1.

5 Experiment

This section demonstrates the effectiveness of feature augmentation based on disentangled representation in debiasing with both quantitative and qualitative evaluation. We compare our method with the previous approaches in debiasing with four different datasets with varied bias ratios. Then, we conduct the ablation study which demonstrates the importance of 1) learning disentangled representation 2) feature augmentation, and 3) scheduling feature augmentation. For the qualitative evaluation, we verify how our approach disentangles the intrinsic features and bias features by visualizing them on 2D embedding space via t-SNE [26] and reconstructing images from them.

5.1 Experiment Details

Baselines Our baselines consist of vanilla network, HEX [7], ReBias [6] and LfF [12]. Vanilla denotes the classification model trained only with the original cross-entropy (CE) loss, without any debiasing strategies. HEX and ReBias explicitly presume the texture of an image as a bias type, while LfF requires no prior knowledge on it.

Datasets As shown in Fig. 2, we use two synthetic datasets (Colored MNIST, Corrputed CIFAR-10) and two real-world datasets (BAR, Biased FFHQ) to evaluate the generalization of debiasing baselines over various domains. Biased action recognition (BAR) [12][3] is a human action recognition dataset which is biased on places. The classes of BAR are 'climbing', 'diving', 'fishing', 'racing', 'throwing', and 'vaulting'. Each action is strongly correlated with its locations (e.g., images of 'fishing' are mostly taken at water surfaces). While the original dataset does not include bias-conflicting samples in the training set, we reorganize the validation and test set and include a partial of bias-conflicting samples in the training set for simulating the various degrees of correlation. Biased FFHQ is curated

[3]CC-BY 4.0

Dataset	Ratio (%)	Vanilla [28]	HEX [7]	ReBias [6]	LfF [12]	Ours
		✗	✓	✓	✗	✗
Colored MNIST	0.5	35.19±3.49	30.33±0.76	**70.47**±1.84	52.50±2.43	65.22±4.41
	1.0	52.09±2.88	43.73±5.50	**87.4**±0.78	61.89±4.97	81.73±2.34
	2.0	65.86±3.59	56.85±2.58	**92.91**±0.15	71.03±2.44	84.79±0.95
	5.0	82.17±0.74	74.62±3.20	**96.96**±0.04	80.57±3.84	89.66±1.09
Corrupted CIFAR-10	0.5	23.08±1.25	13.87±0.06	22.27±0.41	28.57±1.30	**29.95**±0.71
	1.0	25.82±0.33	14.81±0.42	25.72±0.20	33.07±0.77	**36.49**±1.79
	2.0	30.06±0.71	15.20±0.54	31.66±0.43	39.91±0.30	**41.78**±2.29
	5.0	39.42±0.64	16.04±0.63	43.43±0.41	50.27±1.56	**51.13**±1.28
BAR	1.0	35.32±0.46	34.18±0.90	37.02±0.26	48.15±0.93	**52.31**±1.00
	5.0	49.91±0.53	36.26±0.60	53.25±0.63	58.81±2.64	**63.5**±1.47
Biased FFHQ	0.5	56.87±2.69	52.83±0.90	59.46±0.64	62.2±1.0	**63.87**±0.31

Table 2: Image classification accuracy evaluated on unbiased test sets of Colored MNIST, Corrupted CIFAR-10, BAR, and bFFHQ datasets with varying ratio of bias-conflicting samples. We denote whether the model requires a bias type in advance by *cross* mark (not require) and *check* mark (require). Best performing results are marked in bold, while second-best results are denoted with underlines.

from FFHQ dataset [27][4] which contains human face images annotated with their facial attributes. Among them, we select age and gender as the intrinsic and bias attribute, respectively, and construct the dataset with images of high correlation between them. More specifically, most of the females are 'young' (i.e., age ranging from 10 to 29) and males are 'old' (i.e., age ranging from 40 to 59). Therefore, bias-aligned samples which compose the majority of the dataset are young women and old men.

For each dataset, we set the degree of correlation by adjusting the number of bias-conflicting samples among the training dataset. The ratio of bias-conflicting samples are 0.5%, 1%, 2% and 5% for both Colored MNIST and Corrupted CIFAR-10, 1% and 5% for BAR, and 0.5% for BFFHQ, respectively. For evaluation, we construct an *unbiased* test set which includes images without the high correlation existing in the training set.

Implementation Details We use multi-layer perceptron (MLP) with three hidden layers for Colored MNIST, and ResNet-18 [28] for the remaining datasets. To accommodate the disentangled vectors, we double the number of hidden units in the last layer of each network. For the training, we set the batch size of 256 for Colored MNIST and Corrupted CIFAR10, and 64 for BAR and BFFHQ. Bias-conflicting augmentation is scheduled to be applied after 10K iterations for each dataset. We report the averaged accuracy of unbiased test sets over three independent trials with the mean and standard deviation. We include the remaining implementation details in our Appendix.

5.2 Quantitative Evaluation

Comparison on unbiased test sets Table 2 shows the comparisons of image classification accuracy evaluated on the unbiased test sets. In general, our approach demonstrates superior performance in both synthetic and real-world datasets against the baselines with large gaps. Especially, compared to the baselines which do not define the bias types in advance (vanilla [28] and LfF [12]), our approach achieves the state-of-the-art performance across all datasets. This indicates that utilizing the diversified bias-conflicting samples through our augmentation plays a pivotal role in learning debiased representation regardless of the bias types.

Our approach also outperforms the baselines (HEX [7] and ReBias [6]) which are tailored for a specific bias type (e.g., color and texture) in the real-world datasets. We even show superior performance compared to HEX in Colored MNIST without defining the bias type beforehand. While ReBias achieves the best accuracy in Colored MNIST, they utilize BagNet [11] in order to focus on the color bias. Even without using such an architecture, we achieve the second best performance which is comparable to ReBias.

[4] CC-BY-NC-SA 4.0

Disentangle	Augment	Scheduled Augment	Colored MNIST	Corrupted CIFAR10	BAR	bFFHQ
–	–	–	52.09±2.88	25.82±0.33	35.32±0.46	56.87±2.69
✓	–	–	74.03±2.40	27.73±1.02	51.79±1.06	59.4±2.46
✓	✓	–	72.29±3.82	32.81±2.47	49.97±0.66	61.27±3.26
✓	✓	✓	**81.73**±2.34	**36.49**±1.79	**52.31**±1.00	**63.87**±0.31

Table 3: We conduct ablation studies on 1) disentangled representation learning, 2) feature augmentation, and 3) scheduling feature augmentation. Each row indicates the different training settings with *check* mark denoting the settings applied. We average the accuracy of each training over three independent trials.

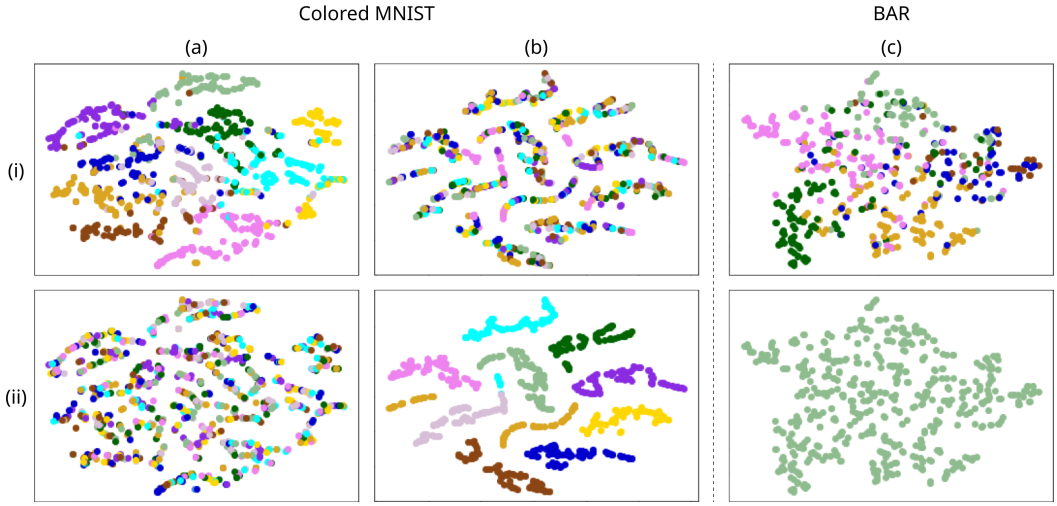

Figure 3: 2D projection of the disentangled representation. z_i and z_b are used for the projection in (a, c), and (b), respectively. We observe that z_i and z_b are well clustered according to the target and bias labels, respectively. Since we do not have access to the bias labels in BAR, we do not show the visualization of z_b with BAR.

Ablation studies Table 3 demonstrates the importance of each module in our approach through ablation studies: 1) disentangled representation learning, 2) feature augmentation, and 3) scheduling feature augmentation. We set the ratio of bias-conflicting samples to 1% for all experiments. We also compare each module with the vanilla network (first row). We observe that performing the scheduled feature augmentation shows the best classification accuracy on the unbiased test sets across all datasets. We also show that performing feature augmentation at the early stage of training does not guarantee the effectiveness of debiasing. For example, in Colored MNIST and BAR, performing feature augmentation at the beginning of training rather aggravates the performance. That is, when the representation of intrinsic attributes and bias attributes are not disentangled at a certain degree, augmented features may generate noisy features. Training with these additional noisy features prevents models from achieving further improvement.

5.3 Analysis

2D Projection of Disentangled Representation Fig.3 shows the projection of latent vectors z_i and z_b extracted from the intrinsic encoder E_i and bias encoder E_b, respectively, on a 2D space. We use Colored MNIST (Fig. 3(a), Fig. 3(b)) and BAR (Fig. 3(c)) for visualizing the disentangled embedding space learned by our method on both synthetic and real-world dataset. We show projection of z_i in Fig. 3(a) and Fig. 3(c) and the projection of z_b in Fig. 3(b). The colors of projected dots in the first row (i) and the second row (ii) indicate the target labels and bias labels, respectively. We observe that z_i are clustered according to the target labels while z_b are clustered with the bias labels. The results represent that our method successfully learns the disentangled intrinsic and bias attributes for both synthetic and real-world datasets.

Accuracy(%)	Colored MNIST		Corrupted CIFAR10		BAR		bFFHQ	
	Intrinsic	Bias	Intrinsic	Bias	Intrinsic	Bias	Intrinsic	Bias
Vanilla	41.63	51.69	24.42	64.29	35.50	–	54.40	45.60
Original	**76.08**	**98.07**	**35.63**	74.16	**65.20**	–	57.40	49.00
Swapping	71.40	94.29	35.14	**76.46**	38.96	–	**58.40**	**51.60**

Table 4: Accuracy from disentangled representation. The ratio of bias-conflicting samples in Colored MNIST, Corrupted CIFAR10, BAR, and bFFHQ are 1%, 1%, 5%, and 0.5%, respectively. We did not evaluate with the bias labels of BAR since it does not have explicit bias labels.

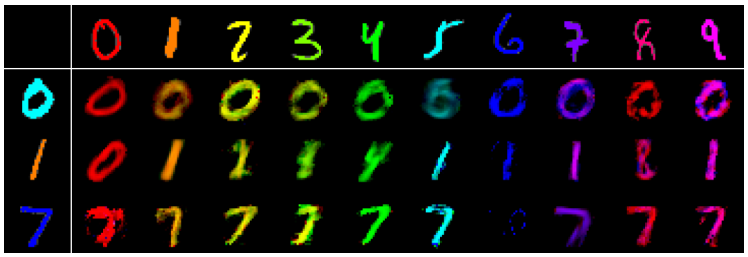

Figure 4: Reconstructed images from disentangled representation in Colored MNIST. Each column and row indicate the samples where the bias attribute (color) and the intrinsic attribute (digit) are extracted, respectively. By swapping the bias features with a given intrinsic feature, we observe that the color changes while maintaining the digit.

Prediction with Disentangled Representation Table 4 shows the test accuracy of C_i and C_b by using $z = [z_i; z_b]$ (original) and $z_{\text{swap}} = [z_i; \tilde{z}_b]$ (swapping) across all unbiased test sets, evaluated with both intrinsic and bias labels. To be more specific, we evaluate the accuracy of predicted labels of 1) $C_i(z)$ and $C_i(z_{\text{swap}})$ with intrinsic labels and 2) $C_b(z)$ and $C_b(z_{\text{swap}})$ with bias labels. We observe that predicting with both disentangled and swapped representation show superior performances compared to vanilla. Both $C_i(z)$ and $C_b(z)$ with the same z achieve high accuracy with intrinsic and bias labels, respectively. This indicates that z_i and z_b are well disentangled, and $C_i(z)$ utilize z_i to predict target labels invariant to the z_b, and vice versa.

Reconstruction of Disentangled Representation Fig. 4 shows the reconstructed images of Colored MNIST by using the disentangled representation of intrinsic features and bias features. Images in the first row and column indicate the images used for extracting the bias attributes (i.e., color) and intrinsic attributes (i.e., digit), respectively. We train an auxiliary decoder by providing the latent vector z from our pre-trained models as input in order to visualize the disentangled representations at the pixel level. By changing the bias attributes (as the column changes), the color of digit changes while maintaining the digit shape. This demonstrates that the bias features and intrinsic features independently contain color and digit information, respectively. Note that the reconstruction loss for updating the decoder is not backpropagated to our pre-trained classification models. Due to this fact, the reconstructed images may lack qualities such as showing blurry images. Since the decoder is trained independently from our method, the reconstructed images may lack qualities such as showing blurry images. Further implementation details are included in our Appendix.

6 Conclusions

In this work, we propose a feature augmentation method based on the disentangled representation of bias features and intrinsic attributes. The main intuition behind our work is that increasing the diversity of bias-conflicting samples beyond a given training set is crucial for debiasing. Since the biased dataset strongly correlates the bias attributes and labels, we intentionally train two different encoders and extract bias features and intrinsic features. After the representations are disentangled to a certain degree, we proliferate the bias-conflicting samples by randomly swapping the vectors. We demonstrate the effectiveness of feature augmentation via extensive experiments, ablation studies, and qualitative evaluation of the disentangled representation. We hope our work inspires future researchers who plan to mitigate the bias via augmentation methods.

References

[1] A. Torralba and A. A. Efros. Unbiased look at dataset bias. CVPR '11, 2011.

[2] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[3] Byungju Kim, Hyunwoo Kim, Kyungsu Kim, Sungjin Kim, and Junmo Kim. Learning not to learn: Training deep neural networks with biased data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[4] Yi Li and Nuno Vasconcelos. Repair: Removing representation bias by dataset resampling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9572–9581, 2019.

[5] 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. *arXiv preprint arXiv:1911.08731*, 2019.

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

[7] Haohan Wang, Zexue He, Zachary L. Lipton, and Eric P. Xing. Learning robust representations by projecting superficial statistics out. In *International Conference on Learning Representations*, 2019.

[8] Remi Cadene, Corentin Dancette, Hedi Ben younes, Matthieu Cord, and Devi Parikh. Rubi: Reducing unimodal biases for visual question answering. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[9] Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. Don't 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, Hong Kong, China, November 2019. Association for Computational Linguistics.

[10] 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 *International Conference on Learning Representations*, 2019.

[11] Wieland Brendel and Matthias Bethge. Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. *International Conference on Learning Representations*, 2019.

[12] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: Training debiased classifier from biased classifier. In *Advances in Neural Information Processing Systems*, 2020.

[13] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[14] Aishwarya Agrawal, Dhruv Batra, and Devi 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, Austin, Texas, November 2016. Association for Computational Linguistics.

[15] Luke Darlow, Stanisław Jastrzębski, and Amos Storkey. Latent adversarial debiasing: Mitigating collider bias in deep neural networks. *arXiv preprint arXiv:2011.11486*, 2020.

[16] Zeyi Huang, Haohan Wang, Eric P. Xing, and Dong Huang. Self-challenging improves cross-domain generalization. In *ECCV*, 2020.

[17] Zhilu Zhang and Mert R Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *arXiv preprint arXiv:1805.07836*, 2018.

[18] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017.

[19] Matthias Minderer, Olivier Bachem, N. Houlsby, and M. Tschannen. Automatic shortcut removal for self-supervised representation learning. In *ICML*, 2020.

[20] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. In *ICLR*, 2021.

[21] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Kumar Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *European Conference on Computer Vision*, 2018.

[22] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018.

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

[24] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[25] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019.

[26] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[27] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See Abstract and Section 1.
   (b) Did you describe the limitations of your work? [Yes] We describe the limitations of our work in the supplementary material.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We describe the potential negative societal impact of our work in the supplementary material.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] No theoretical results are included in the paper.
   (b) Did you include complete proofs of all theoretical results? [N/A] No theoretical results are included in the paper.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We include codes, datasets and instructions attached in the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 5.1 and we include additional implementation details in the supplementary material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Table 1, 2, 3 and Section 5.1.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We include the total amount of compute and the type of resources used in the supplementary material.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We add citations on each dataset in Section 3, Section 3.1, and Section 5.1.
   (b) Did you mention the license of the assets? [Yes] We add footnotes on each dataset in Section 3, Section 3.1, and Section 5.1.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] We check how data was collected and consent was obtained through the open source links of each dataset.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]