Unbiased Classification through Bias-Contrastive and Bias-Balanced Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Datasets for training machine learning models tend to be biased unless the data is collected with complete care. In such a biased dataset, models are susceptible to making predictions based on the biased features of the data. The biased model fails to generalize to the case where correlations between biases and targets are shifted. To mitigate this, we propose Bias-Contrastive (BiasCon) loss based on the contrastive learning framework, which effectively leverages the knowledge of bias labels. We further suggest Bias-Balanced (BiasBal) regression which trains the classification model toward the data distribution with balanced target-bias correlation. Furthermore, we propose Soften Bias-Contrastive (SoftCon) loss which handles the dataset without bias label by softening the pair assignment of the BiasCon loss based on the distance in the feature space of the bias-capturing model. Our experiments show that BiasCon loss and BiasBal regression significantly improve previous debiasing methods in various realistic datasets, respectively, and improve even further when jointly used.

1 Introduction

Machine learning models have achieved extremely high performance in a variety of tasks and domains such as computer vision and natural language processing [14, 7, 34]. Recently, however, many concerns have arisen that such evaluation does not reflect the real-world performance of the model when it is deployed [31, 36]. Among others, failing especially due to the biases existing in the dataset can lead to serious societal side effects such as prejudice or racism beyond simple algorithm failure [40, 30, 5]. If bias features are highly correlated with the object class in the dataset, models tend to use the bias as a cue for the prediction as they are easier to learn but enough to achieve high accuracy even though they are not actually related to the target class [3, 29].

The failure of learning due to the bias existing in the dataset appears in various fields and tasks. To name a few, in image classification task, [16, 27] discovered that state-of-the-art CNNs have texture biases. Visual question answering (VQA) model is also known to be susceptible to biases as it only uses the word occurrence in the question to generate answer [1, 11]. For example, if most of the images of banana in the train sets are yellow, the model directly answers the question "What is the color of the banana?" as "Yellow", without looking at the image of green banana. Surveillance models, which can cause serious societal problems if misclassified, tend to give biased predictions toward sensitive attributes such as race [30].

Considering the importance of the problem, many approaches have been proposed to mitigate such biases in the training dataset. Depending on the presence or absence of bias information, the approaches can be divided into two categories. When the bias label is available, [25, 38] add the bias prediction branch for the unbiased prediction. [35, 10] directly regularize the feature embedding to be indistinguishable across the bias classes. When the bias label is unavailable, [11, 3, 8, 18] design

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

an auxiliary bias-capturing model that learns only bias features and train the main model to learn orthogonal features from those models.

In this paper, we propose two powerful debiasing approaches that can be complementarily applied when the bias label is available. We first propose **Bias-Con**trastive (BiasCon) loss that extends the approaches that directly regularize the feature space by adapting the recent advances of contrastive learning in representation learning. The BiasCon loss utilizes the power of contrastive learning to promote pulling the same target class but different bias class samples closer in the feature space. We further propose the **Bias-Bal**anced (BiasBal) loss that suppresses the utilization of bias features by optimizing toward the data distribution where the target-bias correlation is balanced. Each loss achieves state-of-the-art debiasing performance and shows even higher performance when they are jointly used as they give orthogonal debiasing effects.

To extend to more realistic cases where the bias label is unavailable, we use the observation that the feature space of the bias-capturing model can be used in estimating whether a pair of samples have the same bias features. We thus propose **Soft**en Bias-**Con**trastive (SoftCon) loss that is the BiasCon loss weighted with the cosine distance between samples in the feature space of the bias-capturing model.

We conduct experiments to evaluate the debiasing performance of the proposed methods. For the case where the bias label is available, we evaluate the methods on CelebA [28] and UTKFace [41], which have biases toward sensitive attributes such as gender or race. For the case where the bias label is unavailable, we use ImageNet [32] and ImageNet-A [21] to assess whether the bias of our model has been removed. Our method improves the unbiased accuracy of previous methods by a large margin across all datasets in both with and without bias labels.

Our contributions can be stated as follows:

- We propose a powerful debiasing method, the BiasCon loss, that effectively adapts recent advancements of contrastive learning and the BiasBal loss that further enhances debiasing performance by minimizing the risk toward balanced distribution without bias-target correlation.
- We introduce the SoftCon loss, the extension of the BiasCon loss to the case where the bias label is unavailable, that utilizes the feature space of the bias-capturing model.
- We show that our losses successfully improve the debiasing performance with a large gap in various real-world datasets for both bias label available/unavailable cases.

2 Related work

Bias existing in the data and the vulnerability of machine learning algorithms to such bias has been recently studied as an important problem in various domains and tasks. [31, 36] revealed that various training sets have regularity conditions (e.g., objects should not be occluded) that are unlikely to hold in practical settings, and machine learning models trained on such data fail to generalize in the absence of such conditions. More specifically, [2, 21, 39, 27] showed that state-of-the-art object recognition models are biased toward backgrounds or textures associated with the object class. Natural language processing models are also susceptible to dataset biases. Question Answering (QA) [17, 23, 13, 12] and Visual QA [1, 11] systems make predictions using shortcuts such as the occurrence of specific words in the question rather than the context. Even large language models tend to give biased prediction toward certain gender or race [10]. Meanwhile, the severity of such biased prediction and fairness issues of deployed models are thoroughly investigated in various domains [40, 30, 5]. This brings the importance and necessity of the debiasing method, and recent research has begun to proceed in two main directions depending on the presence or absence of bias labels.

Debiasing when bias label is available An important line of work assumed the situation when the bias factors are categorical attributes (e.g., gender, race) and the meta-information about the bias label of each training data is available. [25] proposed adding a bias prediction branch to the model and train with an additional regularization term that minimizes the mutual information between the bias class and the learned features. [38] suggested to use per-bias classifier head to equalize the effect of bias features. In the risk minimization perspective, [4, 15, 33] proposed distributionally robust optimization that aims the worst-case group distribution shift. [35] devises regularization term based

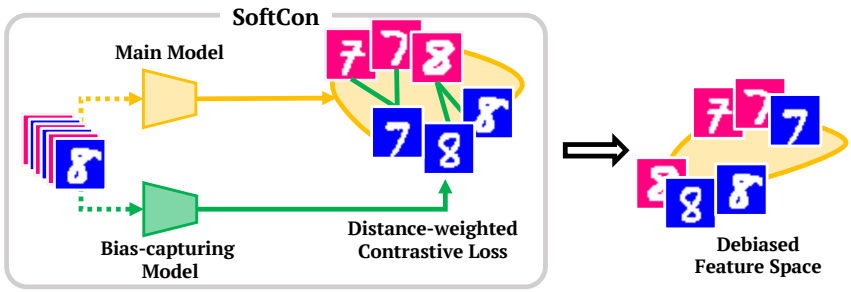


Figure 1: Visualized explanation of our SoftCon loss. As samples with similar bias features are close in the feature space of the bias-capturing model, we design SoftCon loss that does the distance-weighted contrastive learning to pull a pair with the same target class but with different bias features.

on the Gram matrix of features to minimize the entanglement of bias features. For the pretrained language model, [10] appended additional filtering layer and corrects the sentence embedding by maximizing mutual information. We propose a novel regularization term that work upon the success of the contrastive learning framework on self-supervised learning [9, 19, 24] and leverages the bias label in assigning positive pair samples.

Debiasing when the bias label is unavailable It is more realistic to assume that we know the form of biases that have to be removed, but no bias label is available, nor the type of bias is not limited to categorical data (e.g., texture bias of CNNs). Still, there are approaches that mitigate biases in such situations. The first line of research is to utilize the bias-capturing model [11, 3, 8, 18]. The bias-capturing model is an auxiliary model designed to learn bias features. For example, texture biases can be captured by CNNs with small receptive fields [3]. In these approaches, the model is promoted to learn independent features from the bias-capturing models, thus learning debiased features. [29] further relaxed the assumption about the prior knowledge about bias based on the observation that the bias features are learned before the target features. Specific to CNNs for the image classification task, [16, 27] removed texture bias by randomizing the texture of the training data. Our method also utilizes a bias-capturing model, but we propose a more effective way to use the model by considering the distance between training samples in the feature space of the model.

3 Method

This section begins by explaining our problem and setup. Then, we propose two debiasing objectives, **Bias-Con**trastive (BiasCon) loss and **Bias-Bal**ancing (BiasBal) loss, that are effective in situations where the bias is categorical and the bias label is available. We further propose **Soft**en Bias-**Con**trastive (SoftCon) loss which extends the BiasCon loss to more general and challenging situations where the bias is not limited to categorical data and the bias label is not available.

3.1 Problem definition

To formulate our problem, we define a target Y and a bias B that both affects the generation of data X. In such data, we assume that X can be divided into a signal feature X_s explaining the target Y and a bias feature X_b from B. For example, in the data for hair color classification task that is biased toward race (that is, Y=`hair` and B=`race`), X_s includes brown hair feature, and X_b includes white skin feature. Now, the bias in the dataset can be understood as a high correlation between the target class and the bias feature, i.e., the prior $p(Y|B)$ is imbalanced. Then the model may ignore the true signal X_s and instead use X_b in prediction and output $p(Y|X_b)$ as it can still give high accuracy. In the hair color example above, if most white people have blonde hair, the model may output blonde every time it sees the white skin color. Therefore, the biased model fails on the test data distribution with the shifted correlation between Y and B, i.e., $P_{train}(Y, B) \neq P_{test}(Y, B)$.

Our objective is to train the model to make a prediction using the true signal X_s only; that is, the model takes the whole feature X and outputs $p(Y|X_s, X_b) = p(Y|X_s)$ conditionally independent on X_b, so that the model makes indiscriminative predictions across the biases.

3.2 Bias-contrastive loss

We first consider the situation where the bias label is categorical and known for all training samples (e.g., gender or race information of each training sample is available). Here we note the fact that a *biased* model tends to learn features from B and make predictions using those features, hence the samples within the same bias class are close to each other in the feature space. Thus, we devise a loss to explicitly incentivise the model to pull the same target class but *different* bias class sample pairs closer than the other pairs.

Toward this, we borrow and revise the main idea of [24] where supervised contrastive (SupCon) loss pulls the same class samples closer using the contrastive loss and achieves better performance than training solely with the cross-entropy loss. Specifically, we turn the SupCon loss into an effective loss for debiasing, named **Bias**-**Con**trastive (BiasCon) loss, that pulls samples with the same class but different bias closer while pushing the other pairs.

Following the formulation of the SupCon loss, we first construct $2N$ size "multiviewed batch" by applying a pair of randomized augmentation (e.g., random crop, random flip) on N data samples in a single batch: $\{(x_i, y_i, b_i)\}_{i=1}^{2N}$. We then apply the same SupCon loss but to different *positive* pairs (pairs of same target class but different bias class) and formulate the BiasCon loss as follows:

$$L_{\text{BiasCon}} = -\frac{1}{2N} \sum_{i \in I_{mv}} \frac{1}{|J(i)|} \sum_{j \in J(i)} \log \frac{\exp\left(z_i \cdot z_j / \tau\right)}{\sum_{a \in I_{mv}} \exp\left(z_i \cdot z_a / \tau\right)} \tag{1}$$

where $I_{mv} \coloneqq \{1, ..., 2N\}$ is the index set of multiviewed batch, $J(i) \coloneqq \{j \in I_{mv} : y_j = y_i, b_j \neq b_i\}$ is the index set of positive samples paired with i-th sample, $z_i = f(x_i)/\|f(x_i)\|$ is a normalized feature of i-th sample extracted from f, the network up to the penultimate layer, and τ is a temperature hyperparameter.

The final objective is a combination of our BiasCon loss and the standard cross-entropy (CE) loss as follows:

$$L_{\text{CE}} = -\frac{1}{N} \sum_{i \in I} \log h(x_i)[y_i], \tag{2}$$

$$L = L_{\text{CE}} + \alpha \cdot L_{\text{BiasCon}} \tag{3}$$

where $h(x)[y] = p(y|x)$ is the model prediction for class y, $I \coloneqq \{1, ..., N\}$ is the index set of the original batch and α is a weight hyperparameter for the BiasCon loss. Originally, [24] used the SupCon loss solely for the representation learning up to the penultimate layer and additionally trained the fully-connected (FC) layer upon the frozen representation with the CE loss. However in our case, same class but same bias class samples are considered as a negative pairs. Therefore, it is more effective to jointly train with CE loss as it can give signals to incorporate all the same class samples. Thus, unlike [24], we propose to use the BiasCon loss as a regularizer (3). In the Appendix, we empirically show that the joint training with CE loss achieves better performance.

Another specific issue here that contrasts with the SupCon loss is that when $p(b|y)$ is highly imbalanced, a single batch may not contain enough positive pairs because most of the same class samples in the batch will have the same bias features. This may result in unstable optimization. To handle this issue, for training set $\mathcal{D} = \{x_i\}$, we separately sample the mini-batch for the BiasCon loss with a modified sampling frequency $Q(i)$ of i-th sample. We design $Q(i)$ to oversample x_i with low $p(b_i|y_i)$ so that more positive pairs are available in the batch:

$$Q(i) \propto \begin{cases} \frac{1}{p(b_i|y_i)} & \text{if } \frac{1}{p(b_i|y_i)} < \gamma \cdot \min\limits_{y,b} \frac{1}{p(b|y)} \\ \gamma \cdot \min\limits_{y,b} \frac{1}{p(b|y)} & \text{o.w.} \end{cases}. \tag{4}$$

The mini-batch of size N is then sampled as $\{x^{(j)} : x^{(j)} = x_i \text{ where } i \sim Q(i) \text{ for } j = 1, \ldots, N\}$. Note that this formulation clips the largest possible unnormalized $Q(i)$ to be $\gamma \cdot \min Q(i)$ so that every training data is sampled with some chance. We pre-calculate $p(b|y)$ using the Y and B labels in the training set before the training.

It is also instructive to note that EnD [35] has the similar motivation of entangling the features of different bias class samples. However, EnD regularizes the Gram matrix of the feature embedding and thus can be viewed as a truncated version of the BiasCon loss (Explained in the Appendix). Experimental results in Section 4 show the huge performance gap between ours and EnD across all tasks.

3.3 Bias-balanced regression

Highly imbalanced $p(y|b)$ inhibits the learning of features (i.e., x_s) that can correctly classify the minor data under the same bias label. Here, we propose a novel method called bias-balanced regression, which debiases the model by optimizing the model to minimize the risk of the cross-entropy (CE) loss over the data distribution with the uniform correlation between Y and B. Specifically, let $P_{train}(X, Y, B)$ be the data distribution of training set and let P_u be the unbiased test data distribution with the uniform correlation between Y and B. Here, the test data can be understood as the shifted distribution from P_{train}:

$$P_u(X, Y, B) \coloneqq P_{train}(X|Y, B)P_u(Y, B), \tag{5}$$

$$P_u(Y, B) \coloneqq P_u(Y|B)P_{train}(B) = \frac{1}{C} \cdot P_{train}(B), \tag{6}$$

where C is the number of target classes. Biased model fails to generalize to P_u as the correlation between Y and B in P_{train} is not applicable in P_u. To handle this, we aim to use $P_u(X, Y, B)$ in training to optimize the model performance on the data distribution without correlation between Y and B. However, due to the bias in the training data, we cannot directly access the samples of $P_u(X, Y, B)$ and need to achieve this goal from biased P_{train}.

Toward this, consider the model prediction $p(y|x, b)$ and the corresponding Bayesian interpretation:

$$p(y|x, b) = \frac{p(x|y, b)p(y|b)}{p(x|b)}. \tag{7}$$

As can be seen from (7), the bias of training data causes a distributional shift of $p(y|b)$ in the unbiased test phase, resulting in discrepancy of predictive distributions $p(y|x, b)$ during training and test. To correct this discrepancy, we derive the theorem below:

Theorem 1. *(Bias-balanced Regression) Assume the multinomial logistic regression with the model $h(x)[y] = \frac{e^{\eta_y}}{\sum_{y'} e^{\eta_{y'}}}$ where η_y is the logit for class y, and the training data distribution $P_{train}(X, Y, B)$. Let P_u be the data distribution shifted from P_{train} to have the uniform correlation between Y and B. Let $R(h) \coloneqq \mathbb{E}_{P_u}[L_{CE}(h(x), y)]$ be the cross-entropy loss risk associated with h over P_u. Then, $R(h)$ is the same as $\mathbb{E}_{P_{train}}[L(h(x), y, b)]$, where $L(h(x), y, b)$ is defined as follows:*

$$L(h(x), y, b) \coloneqq -\log \frac{\exp\left(\eta_y + p(y|b)\right)}{\sum_{y'} \exp\left(\eta_{y'} + p(y'|b)\right)}. \tag{8}$$

We name the loss function defined in (8) as **Bias**-**Bal**ancing (BiasBal) loss:

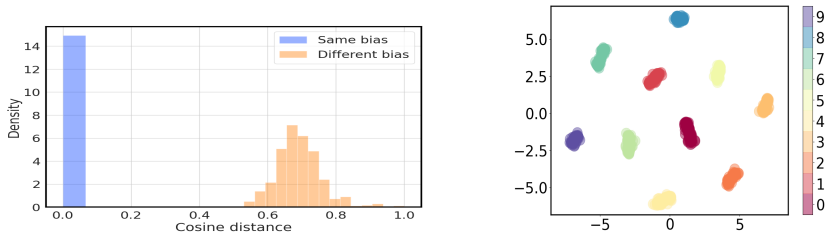
$$L_{BiasBal} \coloneqq \frac{1}{N} \sum_{i \in I} L(h(x_i), y_i, b_i), \tag{9}$$

where I inherits the definition in (2). The BiasBal loss can be used as a replacement for the CE loss, hence it can be seamlessly combined with other existing debiasing methods based on CE loss. In the Appendix, we empirically show that the BiasBal loss further improves the performance of existing methods. Especially, it is well suited with the BiasCon loss, resulting in superior performance in all tasks (Section 4.1).

3.4 Extension to the unknown bias label case

We extend our algorithm to more general settings that do not assume the type of biases to be categorical or the availability of bias labels. However, following [3], we still assume that we have prior knowledge about the form of the bias and we can design a bias-capturing model that inherently make prediction only with x_b, i.e., model predicts $p(y|x_b)$. For example, a CNN model with a small receptive field can only learn local features, thus it only allows to use texture biases to make predictions. (e.g., predict zebra only with stripe features.) We use a pre-trained bias-capturing model as an auxiliary model for debiasing the main model.

To gain information about the bias of the dataset through the lens of the bias-capturing model, we focus on the feature space of the bias-capturing model. Intuitively, when a pair of samples are close to each other in the feature space of the bias-capturing model, it is likely that two samples have similar



(a) Cosine distance distribution

(b) t-SNE of biased features

Figure 2: (a) Distribution of cosine distances between sample pairs of same / different bias classes in the feature space of the bias-capturing model. (b) t-SNE [37] analysis of biased features shows the clear separation between bias classes. Dataset used is a BiasedMNIST [3] with correlation 0.995. Bias-capturing model is a CNN with 1x1 filters and trained for 100 epochs.

bias features in order for the bias-capturing model to work. To check this intuition, we consider a BiasedMNIST [3] dataset, an MNIST [26] dataset with background color highly correlated with each target class. The data of class y has the background color of $b(y)$ with the probability of ρ and has random background color among the rest colors with the probability of $(1-\rho)$ [3]. Here, we use $\rho = 0.995$. We then observe the cosine distance between same/different bias class samples in the feature space of the bias-capturing model. As shown in Figure 2, the same bias class sample pairs have near 0 distances while different bias class sample pairs have far distances. Based on this observation, we propose **Soft**en Bias-**Con**trastive (SoftCon) loss for unknown bias label case that weights the BiasCon loss of each pair with a cosine distance in the feature space of the bias-capturing model as follows:

$$L_{\text{SoftCon}} = -\frac{1}{2N} \sum_{i \in I_{mv}} \frac{1}{C(i)} \sum_{j \in I_{mv} \setminus \{i\}} d_{cos}(w(x_i), w(x_j)) \cdot \log \frac{\exp\left(z_i \cdot z_j / \tau\right)}{\sum_{a \in I_{mv}} \exp\left(z_i \cdot z_a / \tau\right)}, \quad (10)$$

where I_{mv}, z_i and τ inherit the definitions in BiasCon loss (1), $w(x_i)$ is a feature embedding of sample x_i extracted from the penultimate layer of the bias-capturing model, $d_{cos}(u, v) := 1 - \frac{u \cdot v}{\|u\| \|v\|}$ is a cosine distance and $C(i) = \sum_{j \in I_{mv} \setminus \{i\}} d_{cos}(w(x_i), w(x_j))$. With the help of the bias-capturing model, the SoftCon loss in (10) can be utilized without the bias label information, but perhaps someone may be concerned that its performance is highly affected by the behaviors of the bias-capturing model. To address this concern, we evaluate the sensitivity of the SoftCon loss to the performance of the bias capturing model in the Appendix. We find out that the SoftCon loss persists reasonable performance even when the bias-capturing model severely fails to capture the bias features.

The SoftCon loss also suffers from the problem that a single batch may not contain enough positive pairs, i.e., pairs with far cosine distances. However, when the bias label is unavailable, we do not know $p(b|y)$. Instead, we define the bias-sparseness of the sample x of class $y(x)$:

$$t(x) := \sum_{x' \in \mathcal{D}_{y(x)}} d_{cos}(w(x), w(x')), \quad (11)$$

where $\mathcal{D}_y$ is a set of class y samples. If the sample x is bias-sparse, only few same class samples have the same bias, thus more positive pairs are available. Therefore, oversampling the training data based on the bias-sparseness gives the similar effect with oversampling based on $1/p(b|y)$ used in (4). In specific, we define the sampling frequency $Q(i)$ of the i-th sample as follows:

$$Q(i) \propto \begin{cases} t(x_i) & \text{if } t(x_i) < \gamma \cdot \min t(x) \\ \gamma \cdot \min t(x) & \text{o.w.} \end{cases}, \quad (12)$$

and γ is the clipping hyperparameter. In practice, for each class y, we randomly select 1,024 samples and calculate $t(x)$ over those samples.

4 Experiments

We conduct experiments to evaluate how well our proposed method performs debiasing. To do this, we train the model with a biased dataset and measure the debiasing performance with various metrics.

Table 1: Unbiased accuracy evaluated on the BiasedMNIST [3] dataset with various target-bias corrleations. Values with † denotes results directly borrowed from [35].

Corr	Vanilla	LNL [25]	DI [38]	EnD [35]	**BiasCon**	**BiasBal**	**BC+BB**
0.999	10.4	21.7	28.5	52.3†	92.0	76.8	**93.3**
0.997	33.4	59.3	68.3	83.7†	**96.8**	92.0	96.6
0.995	72.1	71.2	81.2	93.9†	**97.3**	94.4	97.1
0.99	89.1	86.8	91.6	96.0†	98.0	96.6	**98.1**
0.95	96.2	96.2	97.3	97.9	98.7	98.2	**98.8**
0.9	97.7	97.8	98.0	98.3	99.0	98.6	**99.1**

The main metric that we focus on is an unbiased accuracy [3, 29]. We first assess BiasCon and BiasBal loss on the dataset where the bias label is categorical and explicitly available. We conduct a controlled experiment on BiasedMNIST [3], where each digit is highly correlated with certain background color. We also evaluate on the real-world dataset CelebA [28] and UTKFace [41] where the dataset is biased toward sensitive attributes (e.g., gender, race). Next, we evaluate the SoftCon loss on BiasedMNIST but without bias label and 9-Class ImageNet [32, 22] that the CNN models tend to show the texture bias. We conduct all experiments with three different seeds and report the average score of each metric. Full results including the standard deviation are available in the Appendix.

4.1 Performance when the bias label is available

We first evaluate our method using datasets with categorical bias and the explicit bias labels. In these datasets, each target class is highly correlated with a certain bias class and the vanilla model tends to use bias features in the prediction. We evaluate on BiasedMNIST [3], CelebA [28] and UTKFace [41]. Note that the bias label is unavailable at the test phase.

Baselines We compare our method with baselines that utilize the bias label in the training time. [25] proposed the framework so called Learning Not to Learn (LNL) which uses additional bias prediction branch to minimize the mutual information between the model feature and the bias label. Domain Independent (DI) training [38] trains separated fully-connected head layer for each bias class and ensembles the output in the final prediction. EnD [35] uses a regularization term to disentangle the features of the same bias class samples.

Controlled experiments The first experiment to validate our method is a controlled experiment using BiasedMNIST dataset proposed by [3]. As explained in Section 3.4, BiasedMNIST is an MNIST [26] dataset with background color highly correlated with each target class with correlation ρ. Vanilla CNN model conditions on the background color to predict the digit as it is easier to learn and achieve high accuracy. We use $\rho \in \{0.9, 0.95, 0.99, 0.995, 0.997, 0.999\}$ to evaluate the effectiveness and robustness of each method in various imbalance ratio.

Table 1 shows the performance of each method in the BiasedMNIST dataset. Our BiasCon loss shows dramatic improvement at every ρ. Especially when the correlation gets very high to 0.999, BiasCon loss still performs debiasing very well, showing an unbiased accuracy of 92.0% while the baselines fail to learn the unbiased features. This shows the effectiveness and robustness of our BiasCon loss. Moreover, the BiasBal loss itself shows much higher performance over the previous state-of-the-art method. This indicates that optimizing toward the unbiased $p(y|b)$ indeed makes the bias features uninformative to the model and prevents learning those features. When two methods are jointly used (BC+BB), the performance gets even better in most cases. The additional improvement implies that two methods are offering two different aspects of debiasing.

Real-world dataset To evaluate our method in the real-world dataset, we consider CelebA [28] and UTKFace [41], as both datasets contain the meta-information about each data. For CelebA, we follow [29] and train the binary classification model to classify the `HeavyMakeup` and `Blonde` attribute that are known to be biased toward `Gender` attribute, i.e., most data with `HeavyMakeup` are `Female`. However, for `Blonde` attribute, we find out that for both `Blonde` and `not-Blonde` classes, `Female` group is a majority group. Therefore, to make the dataset to be biased toward `Gender`, we

Table 2: The unbiased accuracy and bias-conflict accuracy of the model trained on the CelebA [28] dataset with `Gender` as the *bias* attribute.

Target	Acc. Type	Vanilla	LNL [25]	DI [38]	EnD [35]	**BiasCon**	**BiasBal**	**BC+BB**
`Blonde`	Unbiased	82.6	83.1	91.1	88.0	89.5	91.9	**92.2**
	Bias-conflict	66.4	67.6	85.5	78.7	84.6	89.1	**90.1**
`Makeup`	Unbiased	76.5	76.0	76.3	77.8	79.8	77.3	**81.1**
	Bias-conflict	57.1	57.1	57.9	62.0	65.2	61.9	**69.2**

Table 3: The unbiased accuracy and bias-conflict accuracy of the model trained on the UTKFace [41] dataset with `Gender` as the *target* attribute.

Bias	Acc. Type	Vanilla	LNL [25]	DI [38]	EnD [35]	**BiasCon**	**BiasBal**	**BC+BB**
`Race`	Unbiased	87.2	87.5	88.9	88.1	90.4	89.3	**90.9**
	Bias-conflict	79.1	79.6	88.4	81.4	88.9	88.3	**89.5**
`Age`	Unbiased	72.4	73.2	74.9	74.9	79.0	78.3	**79.6**
	Bias-conflict	47.5	49.7	54.6	63.2	75.0	72.1	**76.1**

Table 4: Unbiased accuracy evaluated on the BiasedMNIST [3] dataset without bias labels. Values with † denotes results directly borrowed from [3].

Corr.	Vanilla	LM [11]	RUBi [8]	ReBias [3]	LfF [29]	**SoftCon**
0.999	10.4†	12.1†	13.7†	22.7†	12.1	**86.1**
0.997	33.4†	50.2†	43.0†	64.2†	50.3	**94.7**
0.995	72.1†	78.2†	90.4†	76.0†	89.8	**97.0**
0.99	89.1†	88.3†	93.6†	88.1†	94.2	**97.8**
0.95	96.2	96.1	97.0	97.1	97.4	**98.7**
0.9	97.7	98.3	98.2	98.3	98.1	**99.0**

Table 5: Accuracy on 9-Class ImageNet [22] where bias label is unavailable. Rows with † denotes results directly borrowed from [3].

Acc. Types	Vanilla†	SIN† [16]	LM† [11]	RUBi† [8]	ReBias† [3]	LfF [29]	**SoftCon**
Biased	90.8	88.4	64.1	90.5	91.9	90.6	**93.9**
Unbiased	88.8	86.6	62.7	88.6	90.5	90.6	**93.3**
IN-A	24.9	24.6	15.0	27.7	29.6	28.2	**30.0**

modified the dataset by removing some data from the training set. For UTKFace, we do the binary classification with `Gender` as a target attribute and `Race`, `Age` as bias attributes. We also truncate a portion of data to force the correlation between Y and B to be $p(y|b) = 0.9$. For both datasets, we use ImageNet-pretrained ResNet18 [20]. Details of the experimental setup are available in the Appendix. In addition to the unbiased accuracy, we report the bias-conflict accuracy, which is accuracy on samples with bias features but not a sample of the target class that is highly correlated with that bias.

Results in Table 2 show that our BiasCon and BiasBal loss both outperform the previous methods with large margin. We emphasize that both methods do not hurt the performance of bias-aligning samples as they also increase the total unbiased accuracy. Unlike the results in the controlled experiment, the BiasBal loss shows better performance than the BiasCon loss in the `HairColor` task. We do not claim the superiority of one method over the other between two methods. They give orthogonal effects in debiasing and the joint use of two methods (BC+BB) leads to further improvement. Similarly, our method also shows the superior performance in UTKFace, as shown in Table 3. As a result, the consistently superior results in two real-world datasets imply that both BiasCon and BiasBal loss successfully scale to a large real-world dataset.

4.2 Performance when the bias label is unavailable

We now extend to the case where the bias label is unavailable and the type of bias is beyond categorical data and evaluate our SoftCon loss. Even though the bias label is unavailable, we still assume that we have prior knowledge about the bias and we can design a bias-capturing model [11, 3].

Baselines We compare debiasing methods that work without explicit bias labels. Learned-Mixin+H (LM) [11] and RUBi [8] utilize ensemble learning to promote the model to learn orthogonal features from the bias-capturing model, while ReBias [3] devises a regularizer that induces such learning. LfF [29] observes that bias-conflicting samples have high loss in the early stage of learning and weights such samples. Additionally for the ImageNet, we compare with StylizedImageNet (SIN) [16] that randomizes the texture of ImageNet data to inhibit the model from conditioning on texture biases.

Controlled experiments We use the same configuration with the controlled experiment of the known bias label case. We evaluate on BiasedMNIST [3], but now the bias label is unavailable. For all methods, we design the bias-capturing CNN model with a 1x1 filter. For the SoftCon loss, we pre-train the bias-capturing model for 100 epochs.

In Table 4, we report the performance of each method on BiasedMNIST. Our SoftCon loss shows superior performance with a large margin for all correlation cases. Even for the highest correlation of 0.999, SoftCon loss achieves the unbiased accuracy of 86.1%, improving the previous state-of-the-art method up to 279%. This proves that our distance-based weighting successfully captures the positive pairs, i.e., sample pairs that need to be pulled closer. Our method even outperforms previous bias label *aware* methods in all ρs. This implies that even though the positive pair selection is not perfect, the concept of the bias-contrastive learning is robust enough to give a powerful debiasing effect.

Real-world dataset We test our method on more realistic settings, 9-Class ImageNet [22], a subset of ImageNet [32] with 9 super-classes. Following [3], we aim to train the ResNet18 [20] model from scratch without texture bias which CNN models are susceptible to have. To achieve this, we use BagNet18 [6] as a bias-capturing model since it mainly uses a 1x1 kernel and has small receptive fields. To measure the unbiased accuracy, we use the texture bias label assigned by [3] and average the accuracy measured on each texture group. We also use ImageNet-A (IN-A) [21] as an additional test set as it contains bias-conflicting samples that vanilla CNN models fail to classify correctly.

We report the performance of each method on 9-Class ImageNet in Table 5. Our SoftCon loss successfully perform debiasing while improving the overall (biased) accuracy and ImageNet-A accuracy. Specifically, our method improves unbiased accuracy from 90.5% to 93.3% while improving the biased accuracy and IN-A accuracy 2.18%, 1.35%, respectively. The result indicates that the feature space of the bias-capturing model can give meaningful information about the bias even in the real-world scenario and our SoftCon loss is effective and scalable.

5 Discussion and conclusion

In this work, we proposed the powerful debiasing method named Bias-Contrastive (BiasCon) loss, which effectively revises the contrastive learning framework for debiasing. We further improved the debiasing performance with Bias-Balanced (BiasBal) regression which minimizes the risk of the cross-entropy loss toward unbiased data distribution. To handle the case where the bias label is unavailable, we propose Soften Bias-Contrastive (SoftCon) loss by utilizing the feature space of the bias-capturing model. Experiments on various real-world datasets including CelebA, UTKFace and ImageNet show that our method significantly improves the previous debiasing approaches.

Limitations of our work The first limitation of our work is that, as in [3], our work still assumes that we know the form of the bias (e.g., texture) and we can design the bias-capturing model that mostly uses the bias features in prediction. We believe that we can relax this assumption by considering the features learned earlier as bias features, as in [29]. Furthermore, it is challenging and interesting future work to adapt our model beyond the classification task such as VQA [1, 11].

Societal impact As unbiasedness is the important property that machine learning models need to achieve, we expect our work gives a positive societal impact.

References

[1] Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. Don't just assume; look and answer: Overcoming priors for visual question answering. In *CVPR*, pages 4971–4980. IEEE Computer Society, 2018.

[2] Martín Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *CoRR*, abs/1907.02893, 2019.

[3] 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*, pages 528–539. PMLR, 2020.

[4] Aharon Ben-Tal, Dick den Hertog, Anja De Waegenaere, Bertrand Melenberg, and Gijs Rennen. Robust solutions of optimization problems affected by uncertain probabilities. *Manag. Sci.*, 59(2):341–357, 2013.

[5] Tolga Bolukbasi, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam Tauman Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *NIPS*, pages 4349–4357, 2016.

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

[7] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020.

[8] Rémi Cadène, Corentin Dancette, Hedi Ben-younes, Matthieu Cord, and Devi Parikh. Rubi: Reducing unimodal biases for visual question answering. In *NeurIPS*, pages 839–850, 2019.

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[10] Pengyu Cheng, Weituo Hao, Siyang Yuan, Shijing Si, and Lawrence Carin. Fairfil: Contrastive neural debiasing method for pretrained text encoders. In *International Conference on Learning Representations*, 2021.

[11] Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *EMNLP/IJCNLP (1)*, pages 4067–4080. Association for Computational Linguistics, 2019.

[12] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457, 2018.

[13] Peter Clark, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter D. Turney, and Daniel Khashabi. Combining retrieval, statistics, and inference to answer elementary science questions. In *AAAI*, pages 2580–2586. AAAI Press, 2016.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[15] John C Duchi, Tatsunori Hashimoto, and Hongseok Namkoong. Distributionally robust losses against mixture covariate shifts. *Under review*, 2019.

[16] 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 *ICLR*. OpenReview.net, 2019.

[17] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data. In *NAACL-HLT (2)*, pages 107–112. Association for Computational Linguistics, 2018.

[18] He He, Sheng Zha, and Haohan Wang. Unlearn dataset bias in natural language inference by fitting the residual. In *DeepLo@EMNLP-IJCNLP*, pages 132–142. Association for Computational Linguistics, 2019.

[19] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9726–9735. IEEE, 2020.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778. IEEE Computer Society, 2016.

[21] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *CVPR*, 2021.

[22] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *NeurIPS*, pages 125–136, 2019.

[23] Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems. In *EMNLP*, pages 2021–2031. Association for Computational Linguistics, 2017.

[24] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.

[25] Byungju Kim, Hyunwoo Kim, Kyungsu Kim, Sungjin Kim, and Junmo Kim. Learning not to learn: Training deep neural networks with biased data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9012–9020, 2019.

[26] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[27] Yingwei Li, Qihang Yu, Mingxing Tan, Jieru Mei, Peng Tang, Wei Shen, Alan Yuille, and cihang xie. Shape-texture debiased neural network training. In *International Conference on Learning Representations*, 2021.

[28] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[29] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: Training debiased classifier from biased classifier. *arXiv preprint arXiv:2007.02561*, 2020.

[30] Safiya Umoja Noble. *Algorithms of oppression: How search engines reinforce racism*. nyu Press, 2018.

[31] Jean Ponce, Martial Hebert, Cordelia Schmid, and Andrew Zisserman, editors. *Toward Category-Level Object Recognition*, volume 4170 of *Lecture Notes in Computer Science*. Springer, 2006.

[32] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[33] Shiori Sagawa*, Pang Wei Koh*, Tatsunori B. Hashimoto, and Percy Liang. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2020.

[34] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019.

[35] Enzo Tartaglione, Carlo Alberto Barbano, and Marco Grangetto. End: Entangling and disentangling deep representations for bias correction. *arXiv preprint arXiv:2103.02023*, 2021.

[36] Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In *CVPR*, pages 1521–1528. IEEE Computer Society, 2011.

[37] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[38] Zeyu Wang, Klint Qinami, Ioannis Christos Karakozis, Kyle Genova, Prem Nair, Kenji Hata, and Olga Russakovsky. Towards fairness in visual recognition: Effective strategies for bias mitigation. In *CVPR*, pages 8916–8925. IEEE, 2020.

[39] Kai Yuanqing Xiao, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. Noise or signal: The role of image backgrounds in object recognition. In *International Conference on Learning Representations*, 2021.

[40] Richard S. Zemel, Yu Wu, Kevin Swersky, Toniann Pitassi, and Cynthia Dwork. Learning fair representations. In *ICML (3)*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 325–333. JMLR.org, 2013.

[41] Song Yang Zhang Zhifei and Qi Hairong. Age progression/regression by conditional adversarial autoencoder. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017.

Checklist

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 the contributions written in Section 1.
   (b) Did you describe the limitations of your work? [Yes] See the discussion about the limitations of our work in Section 5
   (c) Did you discuss any potential negative societal impacts of your work? [No] We could not come up with any potential negative societal impact. We instead stated about the positive impact of our work in Section 5.
   (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? [Yes] See the explanation about our method in Section 3 and Appendix.
   (b) Did you include complete proofs of all theoretical results? [Yes] See the Appendix.
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 added our code used for all experiments in the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See the Appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See the Appendix for the standard deviation of every experimental result.
   (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] See the Appendix.
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] All the references of datasets or experimental setup are available in the paper and Appendix.
   (b) Did you mention the license of the assets? [Yes] See the Appendix.
   (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? [N/A]
   (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]