SelecMix: Debiased Learning by Mixing up Contradicting Pairs

Anonymous Authors[1]

Abstract

Neural networks trained with ERM (empirical risk minimization) sometimes learn unintended decision rules, in particular when their training data is biased, i.e. when training labels are correlated with undesirable features. Techniques have been proposed to prevent a network from learning such features, using the heuristic that spurious correlations are "too simple" and learned preferentially during training by SGD. Recent methods resample or augment training data such that examples displaying spurious correlations (a.k.a. bias-aligned examples) become a minority, whereas the other, bias-conflicting examples become prevalent. These approaches are difficult to train and scale to real-world data e.g. because they rely on disentangled representations. We propose an alternative based on mixup that augments the available bias-conflicting training data with convex combinations of existing examples and their labels. Our method, named SelecMix, applies mixup to selected pairs of examples, which show either (i) the same label but dissimilar biased features, or (ii) a different label but similar biased features. To comparing examples along biased features, we use an auxiliary model relying on the heuristic that biased features are learned preferentially during training by SGD. On semi-synthetic benchmarks where this heuristic is valid, we obtain results superior to existing methods, in particular in the presence of label noise, which complicates the identification of bias-conflicting examples.

1. Introduction

The inductive biases contributing to the success of deep neural networks (DNNs) can limit their capabilities for out-of-distribution (OOD) generalization. DNNs are prone to learn preferentially simple, linear predictive correlations from their training data, sometimes ignoring more complex but important patterns (Shah et al., 2020). It has been suggested that simple correlations in the data are often spurious (Dagaev et al., 2021). Consequently, a DNN relying on these spurious correlations will display poor OOD generalization. Spurious correlations in a dataset are often the result of a selection bias, and such datasets are therefore said to be *biased*.

Biased datasets typically contain a majority of so-called *bias-aligned* examples and a minority of *bias-conflicting* ones. In bias-aligned examples, ground truth labels are correlated with both robust and biased features. In bias-conflicting examples, labels are correlated only with robust features. Clearly, the issues of models trained on biased datasets stem from the prevalence of bias-aligned examples. Various approaches have been proposed to train models on biased datasets that ignore biased features. This objective is ill-defined without additional assumptions, or supervision from heterogeneous forms of training data (i.e. non-i.i.d. samples) (Schölkopf et al., 2021).

We study debiased learning with the popular heuristic that spurious correlations and biased features are "easier to learn" than robust ones, meaning that they are incorporated in the model earlier during training by SGD (Shah et al., 2020). This heuristic was previously used for debiasing by Nam et al. (2020); Kim et al. (2021); Lee et al. (2021). See Zhang et al. (2022) for a discussion of the disputed relevance of this heuristic to real-world data.

The heuristic typically serves to train two separate models: (i) an auxiliary model that purposefully relies on biased features, and (ii) the desired debiased model. The auxiliary model serves to guide the training of the debiased one. Nam et al. (2020) train it with a modified objective called the *generalized cross-entropy* (GCE) loss (Zhang & Sabuncu, 2018) to strengthen its reliance on biased, easy-to-learn features. The training of the debiased model either oversamples/upweights the loss of bias-conflicting examples (Nam et al., 2020) or augments the data with novel bias-conflicting examples (Kim et al., 2021; Lee et al., 2021). Augmentation-based methods can be more effective than upweighting and oversampling but they require careful tuning and/or disentangled representations, which makes them difficult to apply

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the SCIS workshop.

to real-world data.

We propose a simple and effective method based on mixup (Zhang et al., 2018) that generates new bias-conflicting examples. Mixup is a general data augmentation that creates convex combinations of randomly-chosen pairs of examples and their labels. A naive application of mixup to a biased dataset is likely to combine bias-aligned examples – the majority of the data – and therefore aggravate the bias issues. Instead, we propose SelecMix, an application of mixup to selected *contradicting pairs* of examples. We define contradicting pairs as having either (i) the same ground truth label but dissimilar biased features, or (ii) different labels but similar biased features. To compare examples with respect to their biased features, we use an auxiliary model. We train this model to implicitly identify the biased features with the heuristic that they are "easier to learn" than robust features. We train this auxiliary model with a novel *supervised contrastive loss* (SC) (Khosla et al., 2020) that amplifies its reliance on easy-to-learn features.

We evaluate our method on standard debiasing benchmarks. SelecMix consistently outperforms prior methods, in particular in the presence of label noise. Label noise is particularly challenging because it increases the difficulty of identifying bias-conflicting examples. We demonstrate this capability on novel versions of standard benchmarks modified to include label noise.

2. Method

We first describe the possible use of mixup as a debiasing strategy, assuming that biased features are precisely identified and that each training example is accompanied with a *bias label* that indicates the value of its biased features. We then move to a more realistic scenario where such labels are unavailable. We implicitly infer bias labels with an auxiliary model that compares examples with respect to these biased features, which we assume to be "easier to learn" than robust ones.

2.1. Mixup for augmenting bias-conflicting examples

We call *bias-aligned examples* the data exhibiting spurious correlations between labels and some biased features. We call *bias-conflicting examples* the remaining part of the data, which does not contain such spurious correlations. Training a model on bias-conflicting examples alone would avoid the bias issues, but these only constitute a small fraction of the training data in the problematic cases that we consider. Our general idea is to use mixup (Zhang et al., 2018) to augment the pool of bias-conflicting examples. Standard mixup is a popular augmentation technique known to improve various measures of robustness and generalization (Zhang et al., 2018; 2020). Mixup builds convex combinations of pairs of

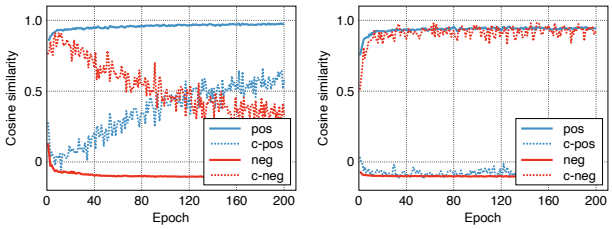


Figure 1. Preliminary experiment on Corrupted CIFAR-10: we train the auxiliary model and examine the similarity of positive, negative, and contradicting pairs during training with the SC loss (**left**) and GSC (**right**). The solid line shows the average cosine similarity of (i) pairs with the same label (*positives*) and (ii) pairs with different labels (*negatives*). The dotted line represents (iii) pairs with the same label but different biased features (*contradicting positives*), and (iv) pairs with different labels but similar biased features (*contradicting negatives*). **Observations:** as training proceeds with the SC loss, the similarity of contradicting positives increases while it decreases for contradicting negatives. In contrast, the proposed GSC loss amplifies the reliance on biased features. The clustering in embedding space remains a good indicator of the similarity of biased features during the whole training process.

examples and their labels. A generated example is given as:

$$\left(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}\right) \leftarrow \left(\lambda\, \boldsymbol{x}_1 + (1-\lambda)\, \boldsymbol{x}_2,\ \lambda\, \boldsymbol{y}_1 + (1-\lambda)\, \boldsymbol{y}_2\right) \quad (1)$$

where (x_1, y_1) and (x_2, y_2) are two original training examples (e.g. image and one-hot label vector) and λ is a random mixing coefficient, $\lambda \sim U[0, 1]$. Our goal is to augment the fraction of bias-conflicting examples in the training data and thereby reduce the reliance of the model on biased features. Assuming for now that bias labels are available, we can generate bias-conflicting examples by applying mixup on the pairs having either (i) the same ground truth label but different bias labels or (ii) different labels but the same bias label. Any such pair includes at least one bias-conflicting example, such that mixup will generate additional ones as long as this original example is assigned a higher mixing weight in Eq.1.

2.2. Replacing bias labels with an auxiliary model

Bias labels are usually not available. We now show how to train an auxiliary model to compare training examples with respect to their biased features, without explicitly identifying these features. We make the assumption that biased features are easier to learn than robust ones, because they are involved in simpler (e.g. linear) predictive patterns. Our auxiliary model is trained to rely primarily on biased features.

We train the auxiliary model with a contrastive objec-

tive (He et al., 2020; Chen et al., 2020; Khosla et al., 2020) because it is known to induce a clustering in embedding space (better than standard cross-entropy) that reflects the similarity of training examples in terms of learned features. These are *biased* features by our assumption, such that the clustering reflects the similarity of examples w.r.t. their (unknown) bias labels. We use the supervised contrastive (SC) loss of Khosla et al. (2020): $\mathcal{L}_{\text{SC}} = -(1/|\mathcal{P}_i|) \sum_{i \in \mathcal{B}} \sum_{k \in \mathcal{P}_i} \log p_{i,k}$ where $p_{i,k} = \frac{\exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_k / \tau)}{\sum_{j \in \mathcal{B} \setminus \{i\}} \exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_j / \tau)}$, $\boldsymbol{z}_i$ is the normalized embedding of image x_i, $\mathcal{B} = \{1, 2, ..., B\}$ is the set of indices in the current mini-batch, $\mathcal{P}_i = \{k \in \mathcal{B} \setminus \{i\} \mid y_i = y_k\}$ is the set of positive examples relative to the example i (i.e. having the same label), and the scalar τ is a temperature hyperparameter.

As an experiment to confirm that the clustering of training examples in embedding space is based on biased features, we train the auxiliary model on all available (biased) data from Corrupted CIFAR-10 dataset with the SC loss and compute the cosine similarity $(\boldsymbol{z}_i \cdot \boldsymbol{z}_j)$ of the embeddings of all pairs of examples. Fig. 1 confirms that the data is clustered according to biased features early in the training. In other words, predictive rules involving biased features are learned faster than those involving robust features, as desired. Since the higher cosine similarity $\boldsymbol{z}_i \cdot \boldsymbol{z}_j$ implies a high probability $p_{i,j}$, we interpret it as the likelihood of the pair (i, j) having similar biased features.

To further amplify the reliance of the auxiliary model on biased features, we define the GSC loss as follows:

$$\mathcal{L}_{GSC} = (-1/|\mathcal{P}_i|) \sum_{i \in \mathcal{B}} \sum_{k \in \mathcal{P}_i} \hat{p}^q_{i,k} \log p_{i,k}, \quad (2)$$

where $\hat{p}^q_{i,k}$ is a *constant* scalar having the same value as $p^q_{i,k}$, meaning that the gradient is not back-propagated through it. The term $\hat{p}^q_{i,k}$ assigns higher weight to sample pairs with a high probability $p_{i,k}$, thus amplifies the reliance on biased features. We further compare the GCE and GSC losses in Appendix A.4.

2.3. Training a debiased model

Equipped with our auxiliary model to quantifying the similarity of training examples in terms of biased features, we can now apply the mixup on pairs having either (i) the same label but dissimilar biased features (*contradicting positives*) or (ii) different labels but similar biased features (*contradicting negatives*).

Contradicting positives. For each instance (x_i, y_i) in the current mini-batch (i.e. the "query"), we pick another one with the lowest cosine similarity (measured in the space of their embeddings produced by the auxiliary model) among the set of positive examples (i.e. with the same label as x_i):

$$k = \underset{j \in \mathcal{P}_i}{\text{argmin}}\ p_{i,j} = \underset{j \in \mathcal{P}_i}{\text{argmin}}\ cos(g_\phi(x_i), g_\phi(x_j)), \quad (3)$$

where $\mathcal{P}_i = \{j \in \mathcal{B} \setminus \{i\} \mid y_i = y_j\}$, $\mathcal{B} = \{1, 2, ..., B\}$ is the set of the sample indices in the current mini-batch, and g_ϕ is the part of our auxiliary model producing embeddings, i.e. $g_\phi(x_i) = \boldsymbol{z}_i$. Since we select the pair among the set of positives, the training loss of the mixed example is $l(\widetilde{x}_i, \widetilde{y}_i) = l(\lambda x_i + (1-\lambda)x_k, \lambda y_i + (1-\lambda)y_k) = l(\lambda x_i + (1-\lambda)x_k, y_k)$. Since most of the examples are bias-aligned in the training set, the query (x_i, y_i) is likely to be bias-aligned. Since x_i and the selected sample x_k have the same label but dissimilar biased features, it is also likely that the biased features of x_k is not correlated to the label. Thus, to effectively generate an example that contradicts the prediction based on biased features, we sample $\lambda \sim U[0, 1]$ and assign the smaller value among λ and $1 - \lambda$ to x_i and the larger one to x_k.

Contradicting negatives. For each query x_i, we select another one with the highest cosine similarity among negative examples (i.e. with a different label) as follows:

$$k = \underset{j \in \mathcal{N}_i}{\text{argmax}}\ p_{i,j} = \underset{j \in \mathcal{N}_i}{\text{argmax}}\ cos(g_\phi(x_i), g_\phi(x_j)), \quad (4)$$

where $\mathcal{N}_i = \{j \in \mathcal{B} \mid y_i \neq y_j\}$. Similarly, we sample $\lambda \sim U[0, 1]$ and let $\lambda \leftarrow \min(\lambda, 1 - \lambda)$. In standard mixup, the training loss of the mixed sample is: $l(\widetilde{x}_i, \widetilde{y}_i) = l(\lambda x_i + (1-\lambda)x_k, \lambda y_i + (1-\lambda)y_k) = \lambda \cdot l(\lambda x_i + (1-\lambda)x_k, y_i) + (1-\lambda) \cdot l(\lambda x_i + (1-\lambda)x_k, y_k)$. However, considering the fact that (i) the query (x_i, y_i) is likely to be bias-aligned, and (ii) the query x_i and the selected sample x_k share similar biased features, the first term $\lambda \cdot l(\lambda x_i + (1-\lambda)x_k, y_i)$ does not contradict the prediction based on biased features. Thus, rather than interpolating the label, we assign $\widetilde{y}_i \leftarrow y_k$.

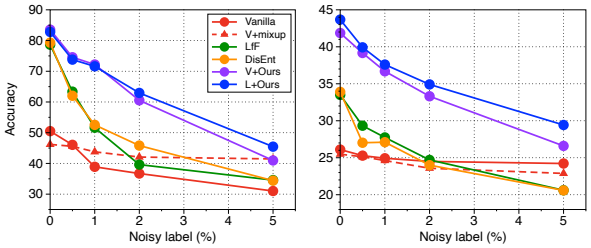
3. Experiments

In this section, we evaluate the proposed method on standard datasets used in the debiasing literature. We also perform experiments on novel versions of these datasets that include label noise. See Appendix A.2 for a description of the datasets and Appendix A.3 for experimental details.

Existing methods. We compare our approach to the prior works *Learning from failure* (LfF) (Nam et al., 2020) and *Disentangled feature augmentation* (DFA) (Lee et al., 2021). Both also rely on the heuristic "easy-to-learn" property of biased features. LfF trains an auxiliary model with the generalize cross-entropy (GCE) loss to amplify its reliance on biased features, then reweights examples for training a debiased model. DFA disentangles biased and robust features with a similar principle as LfF, then augments the data for training a debiased model by swapping the biased

Table 1. Main results. (*): Methods tailored to hard-coded forms of biases (e.g. image texture). (○): Methods using explicit bias labels. (†): Methods using the "easy-to-learn" heuristic. Numbers for HEX are borrowed from (Lee et al., 2021).

Dataset	Ratio (%)	Baseline	HEX *	ReBias *	EnD ○	LfF †	DFA †	Baseline + Ours †	LfF + Ours †
Colored MNIST	0.5	35.71±0.83	30.33±0.76	**71.42**±1.41	56.98±4.85	63.86±2.81	67.37±1.61	70.47±1.66	70.00±0.52
	1.0	50.51±2.17	43.73±5.50	**86.50**±0.97	73.83±2.09	78.64±1.51	80.20±1.86	83.55±0.42	82.80±0.71
	2.0	65.40±1.63	56.85±2.58	**92.95**±0.21	82.28±1.08	84.95±1.71	85.61±0.76	87.03±0.58	87.16±0.62
	5.0	82.12±1.52	74.62±3.20	**96.92**±0.09	89.26±0.27	89.42±0.65	89.86±0.80	91.56±0.17	91.57±0.20
Corrupted CIFAR-10	0.5	23.26±0.29	13.87±0.06	22.13±0.23	22.54±0.65	29.36±0.18	30.04±0.66	38.14±0.15	**39.44**±0.22
	1.0	26.10±0.72	14.81±0.42	26.05±0.10	26.20±0.39	33.50±0.52	33.80±1.83	41.87±0.14	**43.68**±0.51
	2.0	31.04±0.44	15.20±0.54	32.00±0.81	32.99±0.33	40.65±1.23	42.10±1.04	47.70±1.35	**49.70**±0.54
	5.0	41.98±0.12	16.04±0.63	44.00±0.66	44.90±0.37	50.95±0.40	49.23±0.63	54.00±0.38	**57.03**±0.48
BFFHQ	0.5	56.20±0.35	52.83±0.90	56.80±1.56	56.53±0.61	65.60±1.40	61.60±1.97	**71.60**±1.91	70.80±2.95



(a) Colored MNIST (b) Corrupted CIFAR-10

Figure 2. Unbiased accuracy under the presence of label noise. In the training data, the ratio of bias-conflicting samples is fixed at 1% and we report results for a ratios of samples with incorrect labels of $\{0.5\%, 1\%, 2\%, 5\%\}$. The accuracy of the proposed method degrades less than existing methods that also rely on the "easy-to-learn" heuristic. Note that V+mixup denotes the vanilla mixup (Zhang et al., 2018).

features across examples. We also include the *Entangling and disentangling* (EnD) method (Tartaglione et al., 2021), Rebias (Bahng et al., 2020) and HEX (Wang et al., 2018). EnD leverages explicit bias labels. Rebias and Hex are designed for a specific, known form of biased features such as texture in images.

3.1. Main results

We apply our method to a vanilla ResNet18 (Baseline + Ours) as well as on top of the LfF method (LfF + Ours). We evaluate methods on Colored MNIST, Corrupted CIFAR10, and BFFHQ. As shown in Table 1, our method consistently outperforms existing ones, except ReBias on Colored MNIST. Their bias-capturing model BagNet (Brendel & Bethge, 2018)) is specifically tailored to color and texture as biased features by relying on local image patches as input. Both DFA and our method augment the pool of bias-conflicting training examples, but DFA's reliance on disentangled representations seems problematic on the more complex datasets. Our method performs well on all datasets, owing to the simplicity of the mixup strategy. Our method outperforms DFA by a large margin on BFFHQ while the gap is smaller on Colored MNIST where the disentanglement is easier.

3.2. Results under the presence of label noise

Label noise can negatively affect methods relying on the identification of bias-conflicting examples. The reason is that training examples with noisy (i.e. incorrect) labels are difficult to distinguish from the bias-conflicting examples that we wish to upweight. Fig. 2 shows that our method maintains better performance under label noise than competing ones. We hypothesize that the robustness of our method comes from the nature of mixup, which is known to improve robustness (Zhang et al., 2020). Note that we used the same hyperparameters for the label noise experiments and the main experiments.

4. Conclusions

We presented a method for debiased learning that generates augmentations with mixup on selected pairs of training examples. The core element is the selection of these pairs, which uses an auxiliary model trained with a loss specifically designed to amplify its reliance on biased features. The whole approach relies on the heuristic that spurious correlations are "easy to learn" and that biased features are therefore learned early during training by SGD. While this property is disputed (Zhang et al., 2022), our method outperforms existing approaches on semi-synthetic datasets designed to display this property. Unlike existing methods, ours remains effective in the presence of label noise.

References

Bahng, H., Chun, S., Yun, S., Choo, J., and Oh, S. J. Learning de-biased representations with biased representations. In *International Conference on Machine Learning (ICML)*, 2020.

Brendel, W. and Bethge, M. Approximating cnns with

bag-of-local-features models works surprisingly well on imagenet. In *International Conference on Learning Representations*, 2018.

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

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Dagaev, N., Roads, B. D., Luo, X., Barry, D. N., Patil, K. R., and Love, B. C. A too-good-to-be-true prior to reduce shortcut reliance. *arXiv preprint arXiv:2102.06406*, 2021.

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

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020.

Hong, Y. and Yang, E. Unbiased classification through bias-contrastive and bias-balanced learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., and Krishnan, D. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020.

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

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

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

LeCun, Y. and Cortes, C. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

Lee, J., Kim, E., Lee, J., Lee, J., and Choo, J. Learning debiased representation via disentangled feature augmentation. *Advances in Neural Information Processing Systems*, 34, 2021.

Li, Y. and Vasconcelos, N. Repair: Removing representation bias by dataset resampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9572–9581, 2019.

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

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2019.

Schölkopf, B., Locatello, F., Bauer, S., Ke, N. R., Kalchbrenner, N., Goyal, A., and Bengio, Y. Toward causal representation learning. *Proceedings of the IEEE*, 109(5): 612–634, 2021.

Shah, H., Tamuly, K., Raghunathan, A., Jain, P., and Netrapalli, P. The pitfalls of simplicity bias in neural networks. *Advances in Neural Information Processing Systems*, 33: 9573–9585, 2020.

Tartaglione, E., Barbano, C. A., and Grangetto, M. End: Entangling and disentangling deep representations for bias correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13508–13517, 2021.

Wang, H., He, Z., Lipton, Z. C., and Xing, E. P. Learning robust representations by projecting superficial statistics out. In *International Conference on Learning Representations*, 2018.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

Zhang, J., Lopez-Paz, D., and Bottou, L. Rich feature construction for the optimization-generalization dilemma. *arXiv preprint arXiv:2203.15516*, 2022.

Zhang, L., Deng, Z., Kawaguchi, K., Ghorbani, A., and Zou, J. How does mixup help with robustness and generalization? In *International Conference on Learning Representations*, 2020.

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

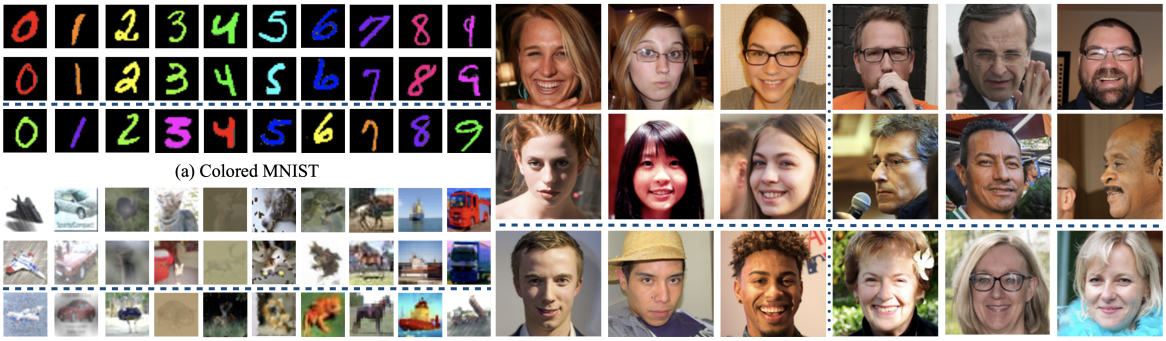
A. Appendix

A.1. Related work

Debiasing when a bias label or a specific form of bias is known. Prior works on debiasing assumed certain information about the bias features and leverage this to train a debiased model from a biased dataset. These approaches are categorized by the type of assumptions about the bias. One category assumes that the explicit bias label is accessible (Hong & Yang, 2021; Tartaglione et al., 2021; Kim et al., 2019; Sagawa et al., 2019; Li & Vasconcelos, 2019). Since the bias label is not always available, another line of work instead assumes a specific form of the bias (e.g., color or texture) and devises an auxiliary bias-capturing model which is designed to intentionally learn the bias features (Wang et al., 2018; Bahng et al., 2020; Cadene et al., 2019). For example, ReBias (Bahng et al., 2020) utilized BagNet (Brendel & Bethge, 2018), which is designed to be intentionally biased toward the texture information, to learn the representation independent of it.

Debiasing biases that are learned more quickly than robust features. The bias features are learned faster than the robust features (Lee et al., 2021; Nam et al., 2020), observing that it degrades the generalization when it is learned earlier (Nam et al., 2020). A popular approach is to train an auxiliary biased model with the Generalized Cross Entropy (GCE) loss (Zhang & Sabuncu, 2018) deliberately learning the easy-to-learn bias. The biased model is used to guide the training of the debiased model, *e.g.,* to identify and upweight the loss of bias-conflicting examples (Nam et al., 2020) or to augment them (Kim et al., 2021; Lee et al., 2021). LfF (Nam et al., 2020) proposed to learn biased and debiased models simultaneously. Based on the relative loss on the biased and debiased model, each example is re-weighted having greater weights to the bias-conflicting examples. On top of their framework, DFA (Lee et al., 2021) emphasized the importance of diversifying bias-conflicting examples and proposed a feature augmentation scheme, which disentangles the bias and robust features and then randomly swaps the bias feature between the examples to generate bias-conflicting examples. While augmentation-based methods (Kim et al., 2021; Lee et al., 2021) require the disentanglement, which is challenging to scale to real-world datasets. We resolve this issue by utilizing a devised mixup to efficiently generate the bias-conflicting examples.

A.2. Description of the datasets



(a) Colored MNIST

(b) Corrupted CIFAR-10

(c) BFFHQ

Colored MNIST is a modified version of MNIST (LeCun & Cortes, 2010). It consists of colored images of ten digits where each digit is correlated with the color (e.g., most images of 1 are colored with red). Target labels are the digit identity (i.e. 0 to 9). The color is the biased feature. **Corrupted CIFAR10** is constructed by applying different types of corruptions to images of different types of objects from CIFAR-10 (Krizhevsky & Hinton, 2009). For example, most images of dogs are corrupted with a Gaussian blur. **Biased FFHQ** (BFFHQ) (Lee et al., 2021) is constructed from the dataset of human faces FFHQ (Karras et al., 2019). The target label in BFFHQ is the age, and the biased feature is the gender, which is a binary annotation in the original dataset.

The ratio of bias-conflicting samples in the training set is $\alpha \in \{0.5\%, 1\%, 2\%, 5\%\}$ in {Colored MNIST, Corrupted CIFAR10} and $\alpha = 0.5\%$ in BFFHQ. All datasets are available in the official repository of DFA (Lee et al., 2021).

A.3. Experimental details

Setup. The backbone architecture used of all methods is a 3-layer MLP for Colored MNIST, and a ResNet18 (He et al., 2016) for Corrupted CIFAR10 and BFFHQ. We set the batch size of 256 for {Colored MNIST, Corrupted CIFAR-10} and 64 for BFFHQ. We train the models for 200 epochs for {Colored MNIST, BFFHQ} and 300 epochs for Corrupted

CIFAR-10. To train the debiased model, we use the Adam optimizer with learning rates of 0.005, 0.001, and 0.0001 for Colored MNIST, Corrupted CIFAR-10, and BFFHQ, respectively. To train the auxiliary model, we use the Adam optimizer with learning rates of 0.01 and 0.001 for Colored MNIST and Corrupted CIFAR-10, respectively and standard SGD with a learning rates of 0.4 for BFFHQ. All experiments were performed with a single Nvidia RTX 3090 GPU.

Implementation details. Regarding the hyperparameters of our method, we fixed the temperature of the contrastive loss at $\tau = 0.2$, and the hyperparameter of the GCE/GSC losses $q = 0.7$ for all datasets. For the baselines, we follow the hyperparameters suggested by the original (Nam et al., 2020; Lee et al., 2021), even if they used different hyperparameter configurations for the different ratios in the same dataset. Since there is no provision for an unbiased validation set in most existing benchmarks, we follow the evaluation protocol of prior work (Bahng et al., 2020; Nam et al., 2020; Kim et al., 2021; Lee et al., 2021) and report the best test set accuracy (i.e. an "oracle" model selection).

Details of experiments with label noise (Sec. 3.2) We modified Colored MNIST ($\alpha = 1\%$) and Corrupted CIFAR-10 ($\alpha = 1\%$) by replacing the label with a random one for a portion $\beta \in \{0.5\%, 1\%, 2\%, 5\%\}$ of the training examples.

A.4. Discussion on the relationship between the GCE and GSC losses

We discuss below the differences between the generalized cross-entropy (GCE) (Zhang & Sabuncu, 2018) and GSC losses. To begin with, we first explain how GCE loss amplifies the reliance of the auxiliary model on biased features, compared to the standard cross-entropy (CE) loss. It is defined as $\mathcal{L}_{\text{GCE}}(\boldsymbol{p}, y) = (1 - \boldsymbol{p}_y^q)/q$ where $\boldsymbol{p}$ is the softmaxed vector of predictions from the model and $\boldsymbol{p}_y$ its yth component, y is the ground truth class ID, and $q \in (0, 1]$ a scalar hyperparameter. The GCE simplifies to the CE loss as $q \to 0$. Assuming that the predictions are produced by a model of parameters $\boldsymbol{\theta}$, the gradients of GCE and CE losses are related as follows: $\frac{\partial}{\partial \theta}\mathcal{L}_{\text{GCE}_\theta}(\boldsymbol{p}, y) = \boldsymbol{p}_y^q \cdot \frac{\partial}{\partial \theta}\mathcal{L}_{\text{CE}_\theta}(\boldsymbol{p}, y)$. Here, the term $\boldsymbol{p}_y^q$ assigns the higher weight to the samples with a high probability $\boldsymbol{p}_y$, thus upweights the "easy" samples and amplifies the reliance on biased features. While the model trained with CE also focuses on biased features since they are learned first, the GCE was shown to be more effective in identifying bias-conflicting examples by Nam et al. (2020).

Similar to the GCE/CE, our GSC loss improves over the SC loss in encouraging the model to rely primarily on biased features. The term $\hat{p}_{i,k}^q$ in Eq. (2) plays the same role as the term $\boldsymbol{p}_y^q$ in the GCE.