SelecMix: Debiased Learning by Contradicting-pair Sampling

Anonymous Author(s)
Affiliation
Address
email

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 "simple" and learned preferentially during training. Recent methods augment training data such that samples displaying spurious correlations (i.e., *bias-aligned* samples) become a minority, whereas the other, *bias-conflicting* samples become prevalent. However, they require sophisticated techniques such as disentanglement with careful tuning, making them challenging to train the model and scale to complex real-world datasets. In this work, we use the mixup, which builds convex combinations of input images and their labels, to augment the bias-conflicting samples. Mainly, we propose a selective mixup scheme, SelecMix, where the mixup is applied to the pairs having (i) the same label but dissimilar biased features, and (ii) different label but similar biased features. To compare samples with respect to the biased features, we propose the bias-amplified contrastive model relying on the heuristic that biased features are learned preferentially during training. Experimental results demonstrate the effectiveness of the proposed method on both the synthetic and real-world datasets. Furthermore, we validate the robustness of our method under noisy labels, which is more realistic, and challenging to identify bias-conflicting samples.

1 Introduction

Deep neural networks (DNNs) have shown remarkable performance in the applications of computer vision [18] and natural language processing [4]. However, the inductive biases contributing to the success of DNNs also limit their capabilities for out-of-distribution (OOD) generalization. They are vulnerable to learning undesirable decision rules based on the spurious features that have strong correlations with the labels, which we refer to as the *dataset bias* [30]. For example, if birds and blue skies frequently appear together in a dataset (i.e., a selection bias in the data), a DNN might associate *bird* labels with blue pixels rather than with the shape of the animal. In the case of test-time distribution shift and unpredictable shortcuts, the performance of the model degrades dramatically [7].

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.[1] 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

[1] A feature is biased if it displays a pattern that is statistically predictive of the labels over the dataset, though not necessarily on every example. For instance, a blue background may be present in most (but not all) images of birds. These images are said to be *bias-aligned*.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

samples. Various approaches for *debiased learning* encourage models trained on biased datasets to ignore biased features. However, the identification of biased features from i.i.d. data is ill-defined, and requires additional assumptions or supervision with heterogeneous (non-i.i.d.) training samples [26].

In this work, we approach debiased learning with the popular heuristic that biased features are "easier to learn" than robust ones, meaning that they are incorporated in the model earlier during training [28]. Existing works that use this heuristic typically 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 [23, 25]. For example, Nam et al. [23] trains the auxiliary model with the *generalized cross-entropy* (GCE) loss [36] that strengthens its reliance on biased, easy-to-learn features. The simultaneous or subsequent training of the debiased model either augments the data with novel bias-conflicting examples [16, 20] or upweights existing ones [23]. Augmentation can be more effective than upweighting but it requires careful tuning and disentangled representations that makes the application to real-world data difficult.

This work proposes a simple and effective approach based on the mixup [34], which implicitly upweights the bias-conflicting samples in the dataset and generates new ones, but does not require any complicated techniques such as disentanglement. Mixup is a data augmentation method that builds convex combinations of input images and their labels. In a biased dataset, applying mixup on randomly-chosen pairs of examples is likely to involve two bias-aligned examples, since they constitute the majority of the data. We argue that this could aggravate a propensity for the prediction based on the biased features. Therefore, we propose a selective mixup strategy coined SelecMix, where the mixup pair is selected so as to contradict the biased prediction of the model. More precisely, SelecMix selects *contradicting pairs*, i.e., pairs having (i) the same label but dissimilar biased features, and (ii) different labels but similar biased features. In the absence of the explicit bias label, we utilize an auxiliary biased model to measure the similarity of the biased features (i.e., *bias similarity*). To properly measure the bias similarity, we propose to train the bias-amplified contrastive model with the modified supervised contrastive loss [14] to further facilitate the reliance on easy-to-learn features.

We evaluate the method on standard benchmarks for debiasing methods. Experimental results suggest that SelecMix with the bias-amplified contrastive model consistently outperforms prior methods. Ablation studies show independent advantages of both (i) SelecMix, which is shown to be effective compared to the other mixup variants, and (ii) the bias-amplified contrastive model, which improves the biased prediction of the biased model and provides a better measure of bias similarity compared to previous biased models. A distinguishing capability of our method is its applicability to cases of label noise. This setting is challenging due to the increased difficulty in identifying bias-conflicting examples. It better mimics real-world conditions in which debiasing methods could be useful. We demonstrate this capability with novel versions of standard benchmarks that include label noise.

2 Related work

Debiasing when a bias label or a specific form of bias is known. Prior works on debiasing assumed certain information about the biased 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 [11, 15, 21, 24, 29]. 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 biased features [1, 5, 32]. For example, ReBias [1] utilized BagNet [3], 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. A number of recent works assume that biased features are learned more quickly than robust ones [20, 23]. A popular approach is to train an auxiliary biased model with the Generalized Cross Entropy (GCE) loss [36] deliberately learning the biased features. The biased model is used to guide the training of the debiased model, e.g., to identify and upweight the loss of bias-conflicting samples [23] or to augment them [16, 20]. LfF [23] proposed to learn biased and debiased models simultaneously. Based on the relative loss on the biased and debiased model, each sample is reweighted having greater weights to the bias-conflicting samples. On top of their framework, DFA [20] emphasized the importance of diversifying bias-conflicting samples and proposed a feature augmentation scheme, which disentangles the biased and robust features and then randomly swaps the biased feature between the samples to generate

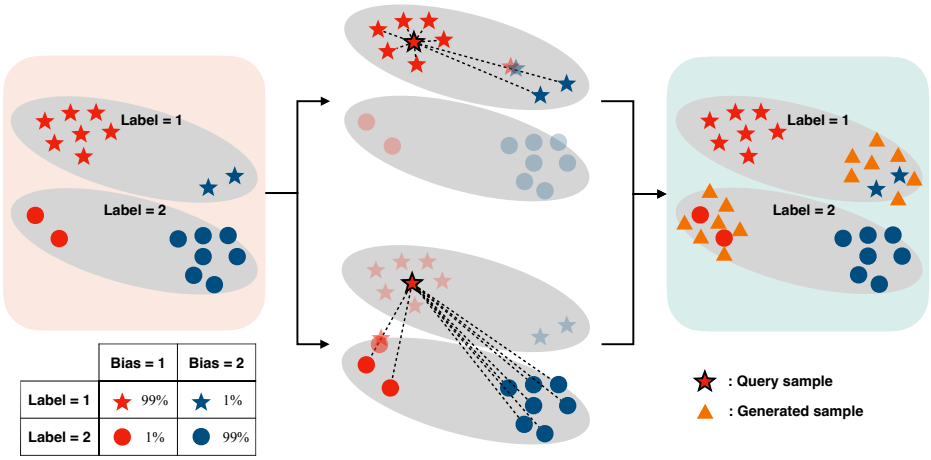


Figure 1: Overview of our method to diversify bias-conflicting samples with the similarity model optimized via our G-SupCon loss (Sec. 3.2). A gray ellipse with the symbols represents a sharing label on the biased embedding space measured by the similarity model. The selective mixup is applied to the pairs of samples (dashed lines) having **(top)** the same label but dissimilar biased features, and **(below)** the different labels but similar biased features (Sec. 3.3). The orange triangles of top and below ellipses represent the generated samples by the previous two procedures, respectively. The legend illustrates the distribution of the exemplary dataset of two labels and two bias labels.

bias-conflicting samples. Since augmentation-based methods [16, 20] require the disentanglement, it is challenging to scale to real-world datasets. We resolve this issue by utilizing a devised mixup to efficiently generate the bias-conflicting samples.

3 Method

We first briefly describe the possible use of mixup as a debiasing strategy, assuming that each training example is provided with a *bias label* that indicates a precise value for its biased features (Sec. 3.1). Then, we move to a more realistic scenario where such labels are unavailable. We introduce the bias-amplified contrastive model which compares examples with respect to biased features (Sec. 3.2) and describe SelecMix, the selective mixup strategy combined with our biased model (Sec. 3.3).

3.1 Mixup for diversifying bias-conflicting samples

Learning from scarce bias-conflicting samples is crucial since they allow us to distinguish between robust features and biased features. In this regard, the recent approaches identify and then upweight their loss [23], and generate bias-conflicting samples [16, 20]. For example, DFA [20] proposes to disentangle the biased and robust feature and randomly swaps the biased features among the samples to generate diverse bias-conflicting samples. The quality of disentanglement is crucial in this approach, especially for the complex real-world images (e.g., to disentangle gender and age features from the images of the human faces). On the contrary, we utilize a simple augmentation strategy, mixup [34]. Mixup interpolates a randomly-chosen pair of two samples and their labels for a generated sample and label, respectively. It is well known to improve the robustness and generalization of the model [34, 35]. A generated sample of the vanilla mixup is defined as follows:

$$(\widetilde{x}_i, \widetilde{y}_i) \leftarrow (\lambda x_i + (1-\lambda)x_k, \lambda y_i + (1-\lambda)y_k), \tag{1}$$

where (x_i, y_i) is the input image and its label, (x_k, y_k) is the other randomly-chosen sample in a batch, and $\lambda \in [0, 1]$ is a mixup coefficient. However, if the ratio of the bias-conflicting samples α in the biased dataset is very low (i.e. $1-\alpha \simeq 1$), then two randomly-chosen samples are highly likely bias-aligned (i.e. $(1-\alpha)^2 \simeq 1$). Since the bias-aligned samples do not contradict the prediction based on the biased features, the samples generated by mixing the pair of bias-aligned samples only strengthen the biased prediction of the model. Therefore, applying random mixup on a highly biased dataset is likely to be ineffective.

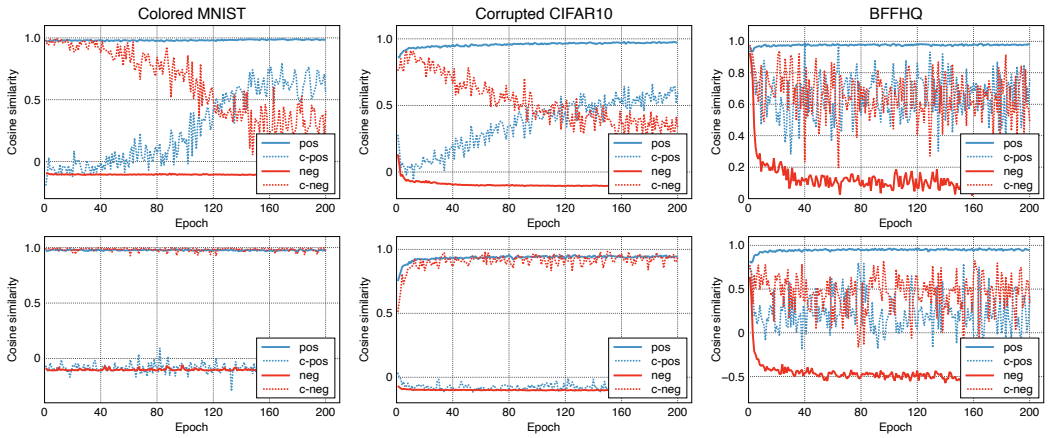


Figure 2: Illustration of the similarity of positive, negative, and contradicting pairs when trained with **(top)** the SupCon and **(bottom)** the G-SupCon (ours) losses. The solid line shows the average cosine similarity of (i) the pairs with the same label (*positives*) and (ii) the pairs with different labels (*negatives*). The dotted line represents (iii) the pairs with the same label but different bias labels (*contradicting positives*), and (iv) the pairs with the different labels but the same bias label (*contradicting negatives*). For the model trained with the SupCon loss, the similarity of contradicting positives increases while decreasing for the contradicting negatives as training proceeds. In contrast, the proposed G-SupCon loss amplifies the reliance on biased features, thus the clustering in embedding space remains a good indicator of the similarity of biased features during the whole training process.

Our goal is to diversify bias-conflicting samples and prevent the model from learning a decision rule based on the biased feature. When the bias label is available, we can generate bias-conflicting samples by applying mixup on the pairs having (i) the same label but different bias labels, and (ii) different labels but the same bias label. However, since the bias label is not always accessible, we utilize the biased model to select the pair. We proceed to elaborate on how to train and leverage the biased model to select the desired mixup pair.

3.2 Replacing the bias labels with the bias-amplified contrastive learning

In the absence of the explicit bias label, we utilize a biased model to quantify the bias similarity of the pair of two samples and selectively choose the mixup pair based on this measure. Exploiting the heuristic that biased features are easier to learn than the robust features, the prior works [20, 23] utilized the Generalized Cross Entropy (GCE) loss [36], $GCE_\theta(p, y) = \frac{1-p_y^q}{q}$, to train the biased model. p and p_y are the softmax output of the model parameterized by θ and its probability assigned to label y, respectively. $q \in (0, 1]$ is a hyperparameter. Note that GCE is a generalization of the cross-entropy (CE) loss in the sense that $GCE_\theta(p, y) \to CE_\theta(p, y)$ for every p and y as $q \to 0$. The GCE loss modifies the gradient of cross-entropy (CE) loss as follows:

$$\frac{\partial}{\partial\theta} GCE_\theta(p, y) = p_y^q \cdot \frac{\partial}{\partial\theta} CE_\theta(p, y). \tag{2}$$

Since the loss of the bias-conflicting samples is relatively high, the GCE loss discourages the model from learning the bias-conflicting samples.

It is known that the contrastive loss [6, 10, 14] more efficiently clusters the samples in the embedding space than the CE loss. To this end, we propose to utilize a contrastive learning framework, which learns hyperspherical embeddings by measuring the similarity of the pairs. Specifically, we utilize the supervised contrastive loss (SupCon) [14] as follows:

$$\mathcal{L}_{SupCon} = \sum_{i \in \mathcal{B}} \frac{-1}{|\mathcal{P}_i|} \sum_{k \in \mathcal{P}_i} \log p_{i,k}, \quad \text{where} \quad p_{i,k} = \frac{\exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}_k / \tau\right)}{\sum_{j \in \mathcal{B} \setminus \{i\}} \exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}_j / \tau\right)}, \tag{3}$$

Algorithm 1 SELECMIX

Input: batch $\mathcal{B} = \{(x_i, y_i)\}_{i=1}^{B}$, biased model g_ϕ
Output: batch $\widetilde{\mathcal{B}} = \{(\tilde{x}_i, \tilde{x}_i)\}_{i=1}^{B}$
Sample $p \sim \text{Uniform}[0, 1]$, $\widetilde{\mathcal{B}} = \{\varnothing\}$
for $i = 1, \cdots, B$ **do**
  Sample $\lambda \sim \text{Beta}[\alpha, \alpha]$
  $\lambda \leftarrow \min(\lambda, 1 - \lambda)$
  if $p > 0.5$: $k = \underset{j \in \mathcal{P}_i}{\text{argmin}}\, cos(g_\phi(x_i), g_\phi(x_j))$
  else: $k = \underset{j \in \mathcal{N}_i}{\text{argmax}}\, cos(g_\phi(x_i), g_\phi(x_j))$
  $(\widetilde{x}_i, \widetilde{y}_i) \leftarrow (\lambda x_i + (1 - \lambda) x_k, y_k)$
  $\widetilde{\mathcal{B}} \leftarrow \widetilde{\mathcal{B}} \cup \{(\widetilde{x}_i, \widetilde{y}_i)\}$
end for

Algorithm 2 Training with the SelecMix

Input: a dataset $\mathcal{D} = \{(x_i, y_i)\}$, a model f_θ, a biased model g_ϕ, the number of iterations T
Output: a debiased model f_θ
Initialize θ and ϕ
for $t = 1, \cdots, T$ **do**
  Draw a batch $\mathcal{B} = \{(x_i, y_i)\}_{i=1}^{B}$ from $\mathcal{D}$
  $\widetilde{\mathcal{B}} = \text{SELECMIX}(\mathcal{B}, g_\phi)$
  Update θ with $\mathcal{L}_{\text{CE}}(\widetilde{\mathcal{B}})$
  Update ϕ with $\mathcal{L}_{\textit{G-SupCon}}(\mathcal{B})$ {Eq. (4)}
end for

where $\mathcal{B} = \{1, 2, \cdots, B\}$ are the sample indices in a batch, $\boldsymbol{z}_i$ is the normalized embedding of image x_i, $\mathcal{P}_i = \{k \in \mathcal{B} \backslash \{i\} \mid y_i = y_k\}$ is the set of positive samples for the sample i, and τ is the temperature of the softmax function.

We train on the biased datasets with the SupCon loss and compute the cosine similarity of the pair to measure the relative similarity. Fig. 2 show that samples are 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. Our observation is aligned with the prior works that neural networks tend to prefer easy-to-learn cues when trained with the CE loss [23, 27].

To further amplify the reliance of the auxiliary model on biased features, we modify the SupCon loss inspired by the GCE loss [20, 23]. The resulting generalized SupCon (G-SupCon) loss is written as follows:

$$\mathcal{L}_{\textit{G-SupCon}} = \sum_{i \in \mathcal{B}} \frac{-1}{|\mathcal{P}_i|} \sum_{k \in \mathcal{P}_i} \hat{p}_{i,k}^{q} \log p_{i,k}, \tag{4}$$

where $\hat{p}_{i,k}^{q}$ indicates the variable detached from $p_{i,k}^{q}$ to be excluded in a gradient-based optimization. Here, the term $\hat{p}_{i,k}^{q}$ impedes the model from learning from bias-conflicting samples and thus stablize learning. Notice that the G-SupCon loss more explicitly facilitates the feature clustering than the GCE loss does by contrastive learning.

3.3 Selective mixup with the biased embedding space

Equipped with the bias-amplified contrastive model in Sec. 3.2 for quantifying bias similarity, we propose to apply the mixup on the pairs having (i) the same label but dissimilar biased features, and (ii) different label but similar biased features. Fig. 1 shows the overview of our proposed method.

Contradicting positives. For each query x_i, we pick the sample with the lowest cosine similarity (i.e., the maximum $l2$-distance in the biased embedding space) among the positives as follows:

$$x_k = \underset{x_j \in \mathcal{P}_i}{\text{argmin}}\; cos(g_\phi(x_i), g_\phi(x_j)), \quad \text{where} \quad \mathcal{P}_i = \{x_j \in \mathcal{B} \setminus \{x_i\} \mid y_i = y_j\}, \tag{5}$$

where g_ϕ is the biased model. Since we select the pair among the positives, the training loss of the mixed sample is $l(\widetilde{x}_i, \widetilde{y}_i) = l(\lambda x_i + (1 - \lambda) x_k, \lambda y_i + (1 - \lambda) y_k) = l(\lambda x_i + (1 - \lambda) x_k, y_k)$. Since most of the samples are bias-aligned in the training set, the query (x_i, y_i) is also likely to be bias-aligned. To effectively contradict the biased prediction of the model, we sample $\lambda \sim \text{Beta}(\alpha, \alpha)$ and assign the smaller value between λ and $1 - \lambda$ to the query x_i and the larger value to x_k.

Contradicting negatives. We propose to select the negative sample that has the highest cosine similarity (i.e., the minimum $l2$-distance) for each query x_i as follows:

$$x_k = \underset{x_j \in \mathcal{N}_i}{\text{argmax}}\; cos(g_\phi(x_i), g_\phi(x_j)), \quad \text{where} \quad \mathcal{N}_i = \{x_j \in \mathcal{B} \mid y_i \neq y_j\}. \tag{6}$$

Similarly, we sample $\lambda \sim \text{Beta}(\alpha, \alpha)$ and let $\lambda \leftarrow \min(\lambda, 1 - \lambda)$. In the vanilla mixup, the training loss of the mixed sample is: $l(\widetilde{x}_i, \widetilde{y}_i) = l(\lambda x_i + (1 - \lambda) x_k, \lambda y_i + (1 - \lambda) y_k) = \lambda \cdot l(\lambda x_i + (1 -$

Table 1: Main results. * denotes the bias-tailored method exploiting the pre-defined forms of bias. ° denotes that the method uses explicit bias label. † utilizes the easy-to-learn heuristic. For HEX, we borrow the numbers from [20].

Dataset	Ratio (%)	Vanilla	HEX * [32]	ReBias * [1]	EnD ° [29]	LfF † [23]	DFA † [20]	V+Ours †	L+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

$\lambda)x_k, y_i) + (1-\lambda) \cdot l(\lambda x_i + (1-\lambda)x_k, y_k)$. However, considering the fact that (i) the query (x_i, y_i) is likely to be bias-aligned, and (ii) the pair (x_i, x_k) shares a 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 the biased features. Thus, rather than interpolating the label, we simply assign $\widetilde{y}_i \leftarrow y_k$. The pseudo-code of the proposed algorithm is presented in Alg. 1 and Alg. 2.

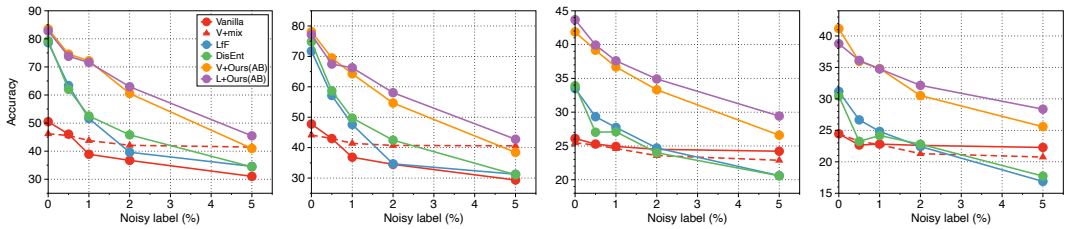
Interpretation. As shown in Fig. 1, our method can be viewed as the augmentation of scarce bias-conflicting samples, by injecting bias-aligned samples as noise to existing bias-conflicting samples since the label of the generated sample is assigned to the label of the selected sample, which is likely to be bias-conflicting. Rather than simply suppressing the weight of the bias-aligned samples, the proposed method effectively leverages abundant bias-aligned samples in the training set. Since bias-conflicting samples are frequently chosen for the mixup pair, our method can be also interpreted as implicitly upweighting the bias-conflicting samples.

4 Experiments

In this section, we validate the effectiveness of the proposed method on standard datasets (Sec. 4.1). In addition, we evaluate our method under the presence of label noise, which is more challenging and realistic, yet less explored (Sec. 4.1). We also provide a detailed analysis of each component of the proposed method, (i) bias-amplified contrastive model (Sec. 4.2) and (ii) selective mixup (Sec. 4.3). We first briefly describe the experimental setup. We provide the experimental details in Appendix A.4.

Datasets. The Colored MNIST is a modified MNIST [19] that consists of the colored images of ten digits where each digit is correlated with the color (e.g., most of the images of "1" is colored with red). Here, the target is a digit (i.e. $0 \sim 9$) and the bias is color. Corrupted CIFAR10 is constructed by applying different types of corruptions to the corresponding objects in the original CIFAR-10 [17] dataset (e.g., most of the images of dog is corrupted with GAUSSIAN BLUR noise). Biased FFHQ (BFFHQ) [20] is constructed based on the real-world dataset FFHQ [13], where the (target, bias) is (age, gender). The ratio of bias-conflicting samples in the training set is $\alpha \in \{0.5\%, 1\%, 2\%, 5\%\}$ for {Colored MNIST, Corrupted CIFAR10} and $\alpha = 0.5\%$ for BFFHQ. All datasets are available in the official repository of DFA [20]. We defer the detailed description of the datasets in Appendix A.2.

Baselines. To evaluate the effectiveness of our method in debiasing, we compare it with the prior methods LfF [23] and DFA [20] which also aim to mitigate the bias which is more easy-to-learn. LfF [23] trains the biased model and debiased model with GCE loss and CE loss, respectively, where the samples are reweighted based on the relative difficulty of the two models when training the debiased model. DFA [20] disentangles the biased and robust feature based on the framework of LfF and swaps the biased feature between samples i.e. feature augmentation. We also compare with the EnD [29] which leverages explicit bias labels, including ReBias [1] and HEX [32], bias-tailored methods exploiting the pre-defined form of bias (i.e., color and texture).



(a) (b) (c) (d)

Figure 3: Unbiased accuracy under the presence of label noise. (a, b) Colored MNIST, (c, d) Corrupted CIFAR-10. In the dataset, the ratio of bias-conflicting samples is $\alpha = 1\%$ and the ratio of noisy label samples is $\{0.5\%, 1\%, 2\%, 5\%\}$. We report both the best accuracy (a, c) and the last test accuracy (b, d). The proposed method shows the robustness toward label noises compared to the prior methods which also utilize the easy-to-learn property of bias.

Table 2: Ablation study of various metrics on the biased embedding space. GT indicates that we used ground truth bias label for SelecMix. cos and l_2 denotes cosine distance and l_2 distance on the embedding space of the biased model trained with GCE loss, respectively. Ours denote the cosine distance on the embedding space of the proposed bias-amplified contrastive model .

SelecMix	GT	cos	l_2	KL-divergence	Ours
Contradicting positives	41.71±0.85	40.31±0.63	38.56±1.25	40.82±0.94	**42.94**±0.44
Contradicting negatives	41.70±0.74	33.69±0.97	34.15±0.50	30.99±0.93	**41.82**±0.98
Both	43.35±0.67	38.05±0.83	38.90±0.96	39.94±1.48	**43.68**±0.51

4.1 Main results

We apply our method to the vanilla ResNet18 (V+Ours) and the LfF (L+Ours) and evaluate these on the Colored MNIST, Corrupted CIFAR10, and BFFHQ. As shown in Table 1, our method consistently outperforms baselines, except for ReBias [1] on the Colored MNIST. Note that their bias-capturing model BagNet [3]) directly accesses the local patches (e.g., color and texture), circumventing learning biased features. While both DFA and ours diversify scarce bias-conflicting samples, DFA's disentanglement seems to struggle on complex real-world datasets. In contrast, our method is less affected by the complexity of the dataset, thanks to the simplicity and effectiveness of the mixup. Specifically, our method outperforms DFA by a large margin on BFFHQ while the gap is relatively small on Colored MNIST, where the disentanglement is much easier.

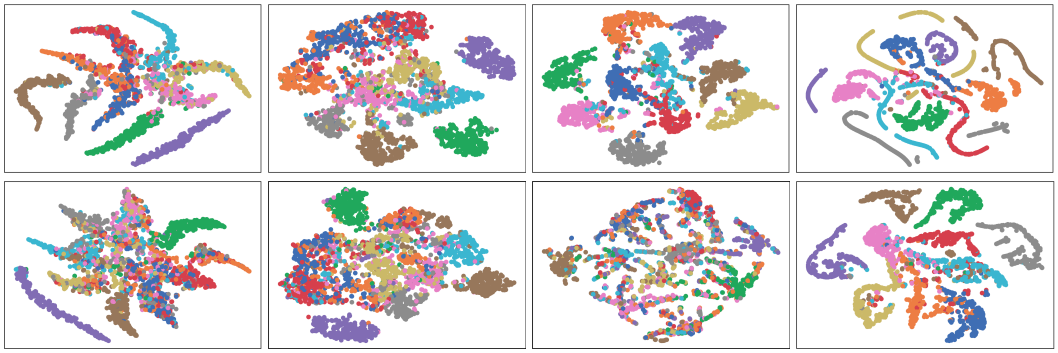
Debiasing under the presence of label noise. The methods that implicitly or explicitly detect the scarce bias-conflicting samples for debiasing are especially vulnerable to the label noises since they identify *noisy samples* and regard these samples as the bias-conflicting samples to amplify. As shown in Fig. 3, our proposed method is relatively robust to label noise compared to the baseline methods. Notably, our method improves the generalization capability of the base model even when there are twice as many noisy label samples as the bias-conflicting samples in the training dataset. We provide the details of the label noise experiments in Appendix A.4.

4.2 Detailed analyses of the bias-amplified contrastive model

Comparison of the biased embedding spaces quantifying bias similarity. We compare the bias-amplified contrastive model g with the biased model of LfF, g_{GCE}, trained with the GCE loss [20, 23] by the various metrics for quantifying the bias similarity. For the metrics, we used l_2 distance, cosine similarity, and the KL-divergence of the softmax outputs of the biased model of LfF. Notice that our method uses cosine similarity. As shown in Table 2, the proposed g achieves the best performance. Especially, the performance gap is significant for the contradicting negatives. This supports our claim in Sec. 3.2 that the contrastive loss effectively clusters the features with the same bias label. Considering that (i) the contrastive loss clusters the samples with the same label and (ii) most of the positive pairs have the same bias label in the biased dataset, it effectively attracts the samples with the same bias label in the embedding space. In contrast, the GCE loss learns the decision boundary w.r.t.

Table 3: Bias prediction accuracy.

Biased model	Corrupted CIFAR-10				BFFHQ
	0.5%	1.0%	2.0%	5.0%	0.5%
LfF [23]	77.44±0.94	73.01±1.70	67.00±0.87	55.58±0.06	51.07±3.06
DFA [20]	79.03±1.15	72.30±0.71	64.71±0.24	52.01±0.70	46.20±1.00
Ours	**95.45**±0.05	**93.39**±0.02	**92.89**±0.06	**87.28**±0.20	**59.13**±0.23



(a) LfF [23] (b) DFA [20] (c) SupCon [14] (d) Ours

Figure 4: The t-SNE visualization of features extracted from (a) the biased model of LfF [23], (b) the biased model of DFA [20], (c) the model trained with the SupCon loss, and (d) the proposed bias-amplified contrastive model. **(Top)** $\alpha = 1\%$, **(bottom)** $\alpha = 5\%$.

the bias labels and does not explicitly cluster the features, thus selecting the contradicting negatives, i.e., the pairs with different labels but a similar biased features, seems to underperform ours.

Bias prediction of the biased model. Table 3 shows the unbiased test accuracy of the bias label prediction of the biased model for each method. The bias-amplified contrastive model does not have a classifier since it is trained with a contrastive loss. We attach a linear classifier on top of the model and fine-tune it. Note that the bias label is used only for the evaluation. As shown in Table 3, the bias-amplified contrastive model shows the best performance. As the ratio of the bias-conflicting samples increases, they degrade the performance. Notice that bias-amplified contrastive model is robust to this effect compared with the prior methods relying on the GCE loss.

Visualization. Fig. 4 illustrates t-SNE [31] plot of the features extracted from the corresponding models trained on the Corrupted CIFAR10. As shown in Fig. 4(d), which corresponds to the proposed bias-amplified contrastive model, we observe that the samples are well clustered on the embedding space of the bias-amplified contrastive model. Similar to the results in Table 3, the GCE-based biased models, i.e., LfF and DFA, suffer from the increasing number of bias-conflicting samples.

4.3 Detailed analysis of the SelecMix

Ablation study. Table 4 shows that the vanilla mixup degrades the generalization capability. While both the contradicting positives (A) and the contradicting negatives (B) significantly contribute to the performance across the benchmarks for both the base models, Vanilla and LfF. Note that we report the accuracy of SelecMix with both of its components, i.e., Ours (AB), for all experiments in the main results (Table 1).

Comparison with a mixup variant when bias label is available. We compare SelecMix with the recently proposed LISA [33], the mixup strategy addressing domain generalization and subpopulation shift problems. LISA similarly applies the vanilla mixup on the pair of (i) the samples with the same label but different domains, and (ii) the samples with different labels but the same domain. Here, the domain label corresponds to the bias label in our experiment. The key differences between SelecMix and LISA are (i) ours does not need the bias label (Sec. 3.2), (ii) we do not use the coefficient λ to interpolate the labels of the generated samples for the reason described in Sec. 3.3, and (iii) we sample λ and let $\lambda \leftarrow \min(\lambda, 1-\lambda)$, i.e., we always assign the higher weight to the selected

Table 4: Detailed analysis of SelecMix. (A) denotes SelecMix with only contradicting positives. (B) denotes SelecMix with only contradicting negatives. (AB) corresponds to the proposed SelecMix.

Dataset	Colored MNIST				Corrupted CIFAR-10				BFFHQ
Ratio (%)	0.5	1.0	2.0	5.0	0.5	1.0	2.0	5.0	0.5
Vanilla	35.71±0.83	50.51±2.17	65.40±1.63	82.12±1.52	23.26±0.29	26.10±0.72	31.04±0.44	41.98±0.12	56.20±0.35
+ mixup	36.90±1.87	46.19±1.93	59.44±3.61	76.28±3.51	23.53±0.80	25.48±0.50	31.88±0.42	43.41±0.56	52.80±0.40
+ Ours (A)	58.49±0.77	74.59±0.77	83.16±1.14	89.32±0.71	37.16±0.09	41.38±0.41	**48.00**±0.64	**54.40**±0.26	66.20±1.80
+ Ours (B)	**71.23**±1.30	81.87±0.94	85.91±0.51	90.53±0.81	35.17±0.47	37.74±0.17	42.77±0.75	49.59±0.05	70.33±0.46
+ Ours (AB)	70.47±1.66	**83.55**±0.42	**87.03**±0.58	**91.56**±0.17	**38.14**±0.15	**41.87**±0.14	47.70±1.35	54.00±0.38	**71.60**±1.91
LfF	63.86±2.81	78.64±1.51	84.95±1.71	89.42±0.65	29.36±0.18	33.50±0.52	40.65±1.23	50.95±0.40	65.60±1.40
+ mixup	44.30±1.03	58.22±2.71	72.44±2.10	85.28±1.39	22.71±0.60	26.32±1.10	32.67±0.46	45.16±0.92	57.53±0.64
+ Ours (A)	57.28±1.96	74.44±0.36	84.20±0.48	90.33±0.22	38.46±0.40	42.94±0.44	49.32±0.28	56.11±0.83	67.60±2.20
+ Ours (B)	**70.26**±1.70	**83.14**±0.86	86.44±0.49	91.49±0.31	37.15±1.31	41.82±0.98	47.01±0.34	53.68±0.02	70.40±2.23
+ Ours (AB)	70.00±0.52	82.80±0.71	**87.16**±0.62	**91.57**±0.20	**39.44**±0.22	**43.68**±0.51	**49.70**±0.54	**57.03**±0.48	**70.80**±2.95

Table 5: Comparison with the mixup variants when the bias label is accessible.

Dataset	Colored MNIST				Corrupted CIFAR-10				BFFHQ
Ratio (%)	0.5	1.0	2.0	5.0	0.5	1.0	2.0	5.0	0.5
Vanilla	35.71±0.83	50.51±2.17	65.40±1.63	82.12±1.52	23.26±0.29	26.10±0.72	31.04±0.44	41.98±0.12	56.20±0.35
+ LISA (A)	48.95±0.75	67.10±0.77	78.28±0.99	87.04±0.72	33.29±0.65	38.62±0.31	45.79±0.13	53.41±0.27	63.20±0.92
+ Ours (A)	58.07±1.94	72.25±0.60	82.35±0.82	90.11±0.14	36.10±0.21	40.55±0.45	46.70±0.73	53.90±0.59	67.67±1.33
+ LISA (B)	52.32±2.30	72.97±1.22	79.74±2.25	86.44±1.10	29.56±0.93	34.23±1.34	40.47±0.67	47.61±0.64	58.93±0.50
+ Ours (B)	72.02±1.39	81.94±1.62	86.04±0.77	88.68±2.32	35.47±0.54	39.27±0.87	43.13±1.29	48.44±0.96	77.40±2.09
+ LISA (AB)	60.85±1.72	74.42±2.27	83.20±0.52	88.73±0.39	32.71±1.09	38.18±0.90	44.15±0.39	51.57±0.45	65.20±0.53
+ Ours (AB)	70.57±2.86	83.38±0.53	87.22±0.70	90.23±0.58	37.02±1.05	41.66±1.10	48.35±0.99	53.47±0.53	75.00±0.53

sample and the lower weight to the query sample (Sec. 3.3). Table 5 empirically confirms the better performance of SelecMix, using the explicit bias label for a fair comparison, over LISA, implying that augmenting the samples close to the bias-conflicting samples is crucial on the biased datasets. More discussions on the comparison between the SelecMix using the explicit bias label and LISA is provided in Appendix A.7.

Table 6: 9-Class ImageNet experiments.

Model	Biased	Unbiased	IN-A
Vanilla	91.27±0.41	90.86±0.42	30.76±0.19
Vanilla + Ours	**92.26**±0.53	**91.98**±0.82	**30.96**±1.37

When equipped with other types of bias capturing model. To validate the compatibility of the selective mixup with other types of bias capturing models, we utilize the BagNet [3] capturing the texture bias. Specifically, we pretrain BagNet for 120 epochs and use this model to measure the bias similarity for our SelecMix. As shown in Table 6, it significantly improves the generalization capability of the vanilla network. We defer the experimental details to Appendix A.5.

5 Discussion and conclusion

In this work, we present the SelecMix for debiasing, where the selective mixup is applied to the pairs of samples having (i) the same label but dissimilar biased features, and (ii) different label but similar biased features. For measuring the bias similarity, we propose the bias-amplified contrastive learning model. Extensive experiments on the extensive benchmarks suggest that our method significantly improves the performance of debiasing. Furthermore, the evaluation in the noisy label environment shows the robustness of our method. **Limitation.** First, our method exploits the easy-to-learn property of bias for training the auxiliary biased model. All biased features may not be learned faster than robust features, although it has not negative effect since the easy-to-learn biases are problematic. Second, the SelecMix requires an auxiliary biased model to quantify the bias similarity, while we can exploit a *pretrained* biased model to reduce the training cost, as we have shown in Appendix. **Societal impact.** If we fail to capture the biases in the auxiliary biased model, it potentially generates the biased output. We may need a supplementary means for critical systems.

References

[1] 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.

[2] Yujia Bao, Shiyu Chang, and Regina Barzilay. Learning stable classifiers by transferring unstable features. In *International Conference on Machine Learning*, pages 1483–1507. PMLR, 2022.

[3] Wieland Brendel and Matthias Bethge. Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. In *International Conference on Learning Representations*, 2018.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[5] Remi Cadene, Corentin Dancette, Matthieu Cord, Devi Parikh, et al. Rubi: Reducing unimodal biases for visual question answering. *Advances in neural information processing systems*, 32, 2019.

[6] 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.

[7] 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.

[8] Giorgio Giannone, Serhii Havrylov, Jordan Massiah, Emine Yilmaz, and Yunlong Jiao. Just mix once: Mixing samples with implicit group distribution. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021.

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

[10] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[11] Youngkyu Hong and Eunho Yang. Unbiased classification through bias-contrastive and bias-balanced learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[12] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *Advances in neural information processing systems*, 32, 2019.

[13] 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*, pages 4401–4410, 2019.

[14] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020.

[15] 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.

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

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

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[19] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

[20] Jungsoo Lee, Eungyeup Kim, Juyoung Lee, Jihyeon Lee, and Jaegul Choo. Learning debiased representation via disentangled feature augmentation. *Advances in Neural Information Processing Systems*, 34, 2021.

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

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

[23] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: De-biasing classifier from biased classifier. *Advances in Neural Information Processing Systems*, 33:20673–20684, 2020.

[24] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2019.

[25] Victor Sanh, Thomas Wolf, Yonatan Belinkov, and Alexander M Rush. Learning from others' mistakes: Avoiding dataset biases without modeling them. In *International Conference on Learning Representations*, 2020.

[26] Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

[27] Luca Scimeca, Seong Joon Oh, Sanghyuk Chun, Michael Poli, and Sangdoo Yun. Which shortcut cues will DNNs choose? a study from the parameter-space perspective. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=qRDQi3ocgR3.

[28] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks. *Advances in Neural Information Processing Systems*, 33:9573–9585, 2020.

[29] Enzo Tartaglione, Carlo Alberto Barbano, and Marco Grangetto. End: Entangling and disentangling deep representations for bias correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13508–13517, 2021.

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

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

[32] Haohan Wang, Zexue He, Zachary C Lipton, and Eric P Xing. Learning robust representations by projecting superficial statistics out. In *International Conference on Learning Representations*, 2018.

[33] Huaxiu Yao, Yu Wang, Sai Li, Linjun Zhang, Weixin Liang, James Zou, and Chelsea Finn. Improving out-of-distribution robustness via selective augmentation. *arXiv preprint arXiv:2201.00299*, 2022.

[34] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[35] Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou. How does mixup help with robustness and generalization? In *International Conference on Learning Representations*, 2020.

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

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]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (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]
   (b) Did you include complete proofs of all theoretical results? [N/A]
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)? [No]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix A
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (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 Appendix A
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]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]
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]