Bias In, Bias Out? Finding Unbiased Subnetworks in Vanilla Models

Supplementary Material

A. Discussion on the mutual information

The mutual information terms $\mathcal{I}(\hat{B}, B)$ and $\mathcal{I}(\hat{Y}, B)$ can be expressed using entropy terms, $\mathcal{H}(\cdot)$, as follows:

$$\begin{aligned} \mathcal{I}(\hat{B}, B) &= \mathcal{H}(B) - \mathcal{H}(B|\hat{B}), \\ \mathcal{I}(\hat{Y}, B) &= \mathcal{H}(B) - \mathcal{H}(B|\hat{Y}). \end{aligned} \tag{A.1}$$

From these,

$$\mathcal{I}(\hat{B}, B) - \mathcal{I}(\hat{Y}, B) = -\mathcal{H}(B|\hat{B}) + \mathcal{H}(B|\hat{Y}). \tag{A.2}$$

In this work, we train a bias classification head $\mathcal{C}_{\text{aux}}$ on top of the encoder $\mathcal{E}$, as specified in Sec. 3.4. If $\mathcal{C}_{\text{aux}}$ perfectly captures the bias encoded in the output of $\mathcal{E}$, *i.e.*, bias classification accuracy $\approx 100\%$, then $\mathcal{H}(B|\hat{B}) \approx 0$. Since the conditional entropy is non-negative, from Eq. (A.2) we have:

$$\mathcal{I}(\hat{B}, B) = \mathcal{I}(\hat{Y}, B) + \mathcal{H}(B|\hat{Y}) \geq \mathcal{I}(\hat{Y}, B). \tag{A.3}$$

As a result, $\mathcal{I}(\hat{B}, B)$ remains an upper bound of $\mathcal{I}(\hat{Y}, B)$, as long as the encoder output contains enough information about the bias B for the classifier $\mathcal{C}_{\text{aux}}$ to leverage. This also motivates periodic updates of the classifier $\mathcal{C}_{\text{aux}}$ to preserve the dependence between $\hat{B}$ and B.

Our setting might be viewed from another perspective. If we assume that our bias classifier $\mathcal{C}_{\text{aux}}$ is (nearly) perfect, which implies $\mathcal{H}(B|\hat{B}) \approx 0$, then we can replace $\hat{B}$ with B in our definition of a biased model in Sec. 3.1. The model can then be considered biased if $\mathcal{I}(\hat{Y}, \hat{B}) \gg \mathcal{I}(Y, B)$. In fact, if the mutual information between the predicted $\hat{Y}$ and $\hat{B}$ is large, while the latter is a perfect prediction of the true B, it is suggested that the classifier $\mathcal{C}$ effectively predicts $\hat{Y}$ from the bias, and minimizing $\mathcal{I}(\hat{Y}, \hat{B})$ would help us to reduce the reliance on bias B. Considering (under the assumption of perfect $\mathcal{C}_{\text{aux}}$) that $\mathcal{I}(\hat{B}, B) = \mathcal{I}(\hat{B}, \hat{B}) = \mathcal{H}(\hat{B})$, we can derive the following statement:

$$\begin{aligned} \mathcal{I}(\hat{Y}, B) &\approx \mathcal{I}(\hat{Y}, \hat{B}) = \mathcal{H}(\hat{B}) - \mathcal{H}(\hat{B}|\hat{Y}) \\ &= \mathcal{I}(\hat{B}, B) - \mathcal{H}(\hat{B}|\hat{Y}) \leq \mathcal{I}(\hat{B}, B). \end{aligned} \tag{A.4}$$

Thus, minimizing $\mathcal{I}(\hat{B}, B)$ is aligned with the goal of minimizing $\mathcal{I}(\hat{Y}, B)$.

Regarding theoretical guarantees, in Eq. (5), $\mathcal{I}(\hat{B}, B)$ is implicitly assumed to be an upper-bound to $\mathcal{I}(\hat{Y}, B)$. This condition is true when $\mathcal{H}(B|\hat{B}) = 0$, for which it is sufficient that $\mathcal{C}_{\text{aux}}$ is a perfect classifier ($\hat{B} \equiv B$), as discussed above; the last condition, however, cannot be always ensured.

B. Reweighting of the cross-entropy loss

We assume that a model that has been trained by traditional empirical risk minimization (ERM) is susceptible to bias when there is a significant prevalence of bias-aligned samples in the training data [11, 27, 30]. We consider a dataset with C target classes $y \in \mathcal{Y}$ and C bias classes $b \in \mathcal{B}$, where each y is strongly correlated with one b. We denote by b_y the bias class correlated with the target class y. To account for bias during training, we modify the cross-entropy loss function $\mathcal{L}_{\text{CE}}(\hat{y}, y)$ by *upweighting* the bias-conflicting group.

Let ρ be the proportion of bias-aligned samples in the training set $\mathcal{D}_{\text{train}}$. We assume that the bias-conflicting samples in $\mathcal{D}_{\text{train}}$ are uniformly distributed across the pairs $\{(y, b) : y \in \mathcal{Y}, b \in \mathcal{B}, b \neq b_y\}$ – as is the case in the BiasedMNIST [1] and Corrupted-CIFAR10 [7] datasets. Inspired by reweighting based on the inverse frequency of the groups [11, 27], we assign weight $r^{\parallel} = \frac{k^{\parallel}}{\rho}$ to bias-aligned samples, and $r^{\perp} = \frac{k^{\perp}}{1-\rho}$ for bias-conflicting samples, where $k^{\parallel}$ and $k^{\perp}$ are constant factors w.r.t. ρ. Let r_j denote the weight assigned to the j-th sample. To find $k^{\parallel}$ and $k^{\perp}$, we impose two constraints:

(*i*) $\sum_{j=1}^{|\mathcal{D}_{\text{train}}|} r_j = |\mathcal{D}_{\text{train}}|$ (normalization), which can be rewritten as

$$\begin{aligned} & r^{\parallel}\rho|\mathcal{D}_{\text{train}}| + r^{\perp}(1-\rho)|\mathcal{D}_{\text{train}}| = |\mathcal{D}_{\text{train}}| \\ \iff & r^{\parallel}\rho + r^{\perp}(1-\rho) = 1 \iff k^{\parallel} + r^{\perp} = 1. \end{aligned} \tag{B.1}$$

(*ii*) If the dataset is unbiased, all samples should receive the same weight, *i.e.*: $\rho = \frac{1}{C} \implies r^{\parallel} = r^{\perp}$, from where we write

$$\frac{k^{\parallel}}{1/C} = \frac{k^{\perp}}{1-1/C} \iff k^{\perp} = (C-1)k^{\parallel}. \tag{B.2}$$

From Equations (B.1) and (B.2), we obtain $k^{\parallel} = \frac{1}{C}$ and $k^{\perp} = \frac{C-1}{C}$. Hence: $r^{\parallel} = \frac{1}{C\,\rho}$ and $r^{\perp} = \frac{C-1}{C\,(1-\rho)}$.

Choice of weights for the datasets considered. In BiasedMNIST and Corrupted-CIFAR10, we have $C = 10$, therefore: $r^{\parallel} = \frac{1}{10\,\rho}$ and $r^{\perp} = \frac{9}{10\,(1-\rho)}$.

In the training set of CelebA [18], we observe an imbalance in terms of distribution of the target class (`BlondHair`) [8, 24]. In effect, most samples are labeled as dark-haired. For that reason, while reweighting the loss, we wish to take into consideration not only the spurious correlation between the gender and the hair color, but also the aforementioned class imbalance. Let c_{B} denote the proportion of blond-haired samples in the training set. Let ρ_{B}

represent the proportion of women inside the blond-haired group, and let ρ_{D} be the proportion of women inside the dark-haired group. Following the idea of loss reweighting based on inverse-frequency of the groups [11, 27], we denote by

$$\begin{cases} r^{\mathrm{MD}} = \frac{k^{\mathrm{MD}}}{(1-\rho_{\mathrm{D}})\,(1-c_{\mathrm{B}})}\,, \\ r^{\mathrm{WD}} = \frac{k^{\mathrm{WD}}}{\rho_{\mathrm{D}}\,(1-c_{\mathrm{B}})}\,, \\ r^{\mathrm{MB}} = \frac{k^{\mathrm{MB}}}{(1-\rho_{\mathrm{B}})\,c_{\mathrm{B}}}\,, \\ r^{\mathrm{WB}} = \frac{k^{\mathrm{WB}}}{\rho_{\mathrm{B}}\,c_{\mathrm{B}}}\,, \end{cases} \tag{B.3}$$

the weights assigned for samples identified as dark-haired men, dark-haired women, blond-haired men, and blond-haired women, respectively, where k^{MD}, k^{WD}, k^{MB} and k^{WB} are constant factors with respect to ρ_{D}, ρ_{B} and c_{B}.

Similarly to the reasoning that led to Equations (B.1) and (B.2), we set the following constraints for CelebA:

(*i*) $\sum_{j=1}^{|\mathcal{D}_{\text{train}}|} r_j = |\mathcal{D}_{\text{train}}|$, which is equivalent to

$$\begin{aligned} &r^{\mathrm{MD}}(1-\rho_{\mathrm{D}})\,(1-c_{\mathrm{B}}) + r^{\mathrm{WD}}\rho_{\mathrm{D}}\,(1-c_{\mathrm{B}}) + \\ &\quad + r^{\mathrm{MB}}(1-\rho_{\mathrm{B}})\,c_{\mathrm{B}} + r^{\mathrm{WB}}\rho_{\mathrm{B}}\,c_{\mathrm{B}} = 1 \\ &\iff k^{\mathrm{MD}} + k^{\mathrm{WD}} + k^{\mathrm{MB}} + k^{\mathrm{WB}} = 1. \end{aligned} \tag{B.4}$$

(*ii*) $\rho_{\mathrm{D}} = \frac{1}{2} \implies r^{\mathrm{MD}} = r^{\mathrm{WD}}$, which in turn can be rewritten (from Eq. (B.3)) as

$$\frac{k^{\mathrm{MD}}}{(1-\frac{1}{2})\,(1-c_{\mathrm{B}})} = \frac{k^{\mathrm{WD}}}{\frac{1}{2}\,(1-c_{\mathrm{B}})} \iff k^{\mathrm{MD}} = k^{\mathrm{WD}}. \tag{B.5}$$

(*iii*) $\rho_{\mathrm{B}} = \frac{1}{2} \implies r^{\mathrm{MB}} = r^{\mathrm{WB}}$, which (from Eq. (B.3)) is equivalent to

$$\frac{k^{\mathrm{MB}}}{(1-\frac{1}{2})\,c_{\mathrm{B}}} = \frac{k^{\mathrm{WB}}}{\frac{1}{2}\,c_{\mathrm{B}}} \iff k^{\mathrm{MB}} = k^{\mathrm{WB}}. \tag{B.6}$$

(*iv*) $c_{\mathrm{B}} = \frac{1}{2} \implies r^{\mathrm{MD}}(1-\rho_{\mathrm{D}})\,(1-\frac{1}{2}) + r^{\mathrm{WD}}\rho_{\mathrm{D}}\,(1-\frac{1}{2}) = r^{\mathrm{MB}}(1-\rho_{\mathrm{B}})\,\frac{1}{2} + r^{\mathrm{WB}}\rho_{\mathrm{B}}\,\frac{1}{2}$ (*i.e.*, if the training set contains the same amount of dark-haired and blond-haired samples, then the total weight assigned to the dark-hair group should be the same as the total weight of the blond-haired group). The second member of the implication is equivalent to

$$\begin{aligned} &r^{\mathrm{MD}}(1-\rho_{\mathrm{D}}) + r^{\mathrm{WD}}\rho_{\mathrm{D}} = r^{\mathrm{MB}}(1-\rho_{\mathrm{B}}) + r^{\mathrm{WB}}\rho_{\mathrm{B}} \\ &\iff k^{\mathrm{MD}} + k^{\mathrm{WD}} = k^{\mathrm{MB}} + k^{\mathrm{WB}}. \end{aligned} \tag{B.7}$$

Solving the system defined by Equations B.4, B.5, B.6 and B.7 leads to $k^{\mathrm{MD}} = k^{\mathrm{WD}} = k^{\mathrm{MB}} = k^{\mathrm{WB}} = \frac{1}{4}$. Finally, by substituting these constants in Eq. (B.3), we obtain

$$\begin{cases} r^{\mathrm{MD}} = \frac{1}{4\,(1-\rho_{\mathrm{D}})\,(1-c_{\mathrm{B}})}\,, \\ r^{\mathrm{WD}} = \frac{1}{4\,\rho_{\mathrm{D}}\,(1-c_{\mathrm{B}})}\,, \\ r^{\mathrm{MB}} = \frac{1}{4\,(1-\rho_{\mathrm{B}})\,c_{\mathrm{B}}}\,, \\ r^{\mathrm{WB}} = \frac{1}{4\,\rho_{\mathrm{B}}\,c_{\mathrm{B}}}. \end{cases} \tag{B.8}$$

In Multi-Color MNIST [17], each image is associated with two bias labels: the left color $b_{\mathrm{L}} \in \mathcal{B}_{\mathrm{L}}$, and the right color $b_{\mathrm{R}} \in \mathcal{B}_{\mathrm{R}}$, with $|\mathcal{B}_{\mathrm{L}}| = |\mathcal{B}_{\mathrm{R}}| = C$. We assume that each bias label is individually correlated with the target label as before, in the single-bias case. Let ρ_{L} and ρ_{R} denote the proportion of bias-aligned samples w.r.t. b_{L} and b_{R}, respectively. We follow [21] and assign the weights

$$\begin{cases} r^{\parallel,\parallel} = \frac{1}{\rho_{\mathrm{L}}\,\rho_{\mathrm{R}}}\,, \\ r^{\parallel,\perp} = \frac{1}{\rho_{\mathrm{L}}\,(1-\rho_{\mathrm{R}})}\,, \\ r^{\perp,\parallel} = \frac{1}{(1-\rho_{\mathrm{L}})\,\rho_{\mathrm{R}}}\,, \\ r^{\perp,\perp} = \frac{1}{(1-\rho_{\mathrm{L}})\,(1-\rho_{\mathrm{R}})}\,, \end{cases} \tag{B.9}$$

to the samples in the groups (*aligned, aligned*), (*aligned, conflicting*), (*conflicting, aligned*), and (*conflicting, conflicting*), respectively, where the first position of the pair indicates whether the group is bias-aligned w.r.t. the label b_{L}, and the second position indicates whether it is bias-aligned w.r.t. b_{R}.

In CivilComments, the data is divided into four groups, according to the pairs (y, b), and the loss reweighting is analogous to the one applied to CelebA.

C. Description of the datasets

We present here an extended description of the datasets employed for the experiments.

BiasedMNIST. Proposed by Bahng *et al.* [1], it consists of a synthetic dataset built upon the classic MNIST dataset [15] by adding to the background of each image a color which is correlated with the target label in the following manner. Each digit $y \in \mathcal{Y} = \{0, 1, 2, \ldots, 9\}$ is associated with a unique *predominant* background color $b_y \in \mathcal{B}$, where $|\mathcal{B}| = 10$, according to a "one-to-one" relation: for each image labeled as digit y, the background is colored with b_y with probability $\rho \in [0, 1]$, and with a random color $b \in \mathcal{B}\backslash\{b_y\}$ with probability $1 - \rho$. Hence, the higher ρ, the stronger is the correlation between digits and colors. As commonly done in the literature for model debiasing, we build the training set $\mathcal{D}_{\text{train}}$ with large ρ, while the unbiased test set $\mathcal{D}_{\text{test}}$ is built with $\rho = 0.1$. Fig. C.1 shows some samples from the BiasedMNIST dataset.

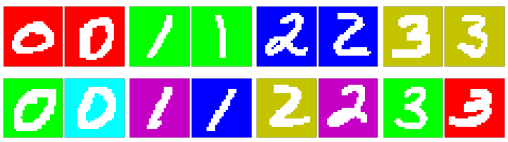

Figure C.1. Samples from the BiasedMNIST dataset [1]. In the first row, bias-aligned images; in the second row, bias-conflicting images.

CelebA. This real-world dataset, built by Liu *et al.* [18],

is composed of 202 599 face images, each described with 40 binary attributes. In CelebA, there exists a spurious relation between the perceived "gender" and the "hair color" attribute, with the majority of individuals with blond hair being labeled as women, and only a small percentage as men. In our experiments, we consider the `BlondHair` attribute as our target, with the attribute `Male` as the bias, similarly to [22, 27, 30]. In Fig. C.2, we show samples corresponding to the four different combinations of the binary attributes `BlondHair` and `Male`.

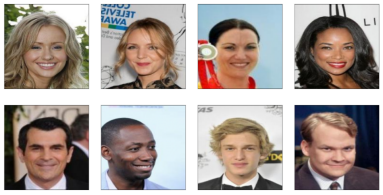

Figure C.2. Samples from the CelebA dataset [18].

Corrupted-CIFAR10. Developed by Hendrycks and Dietterich [7], this dataset is synthesized from the original CIFAR10 data [14], by applying to each image a *corruption* (*e.g.*, brightness, contrast, fog, etc.), and the type of corruption is correlated with the corresponding object class. In Fig. C.3, we show samples from each class in the dataset. As in BiasedMNIST, we denote by ρ the proportion of bias-aligned samples in a set, and we consider $\rho \in \{0.95, 0.98, 0.99, 0.995\}$ for the training set, while the test data is unbiased ($\rho = 0.1$). For each image, the target label y is the object class, while the bias b is the corruption that has been applied.

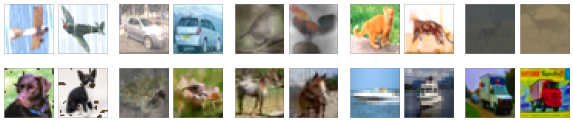

Figure C.3. Samples from the Corrupted-CIFAR10 dataset [7]. We show two images belonging to each class.

Multi-Color MNIST. When more than one bias is present, model debiasing becomes particularly challenging, as mitigating the reliance on one attribute may increase the dependency on another [11]. To evaluate our method in such challenging cases, we employ *Multi-Color MNIST* [17], another dataset synthesized from MNIST. In this dataset, the background of each image consists of two colors: b_{L} on the left side and b_{R} on the right, which are correlated with the digit y according to ρ_{L} and ρ_{R}, respectively (see Fig. C.4). We use $(\rho_{\text{L}}, \rho_{\text{R}}) = (0.99, 0.95)$ in the training set, and $(\rho_{\text{L}}, \rho_{\text{R}}) = (0.1, 0.1)$ in the test set, as in [11, 17]. When applying BISE on Multi-Color MNIST, we employ *two* auxiliary classifiers: one, $\mathcal{C}^{\text{L}}_{\text{aux}}$, to predict b_{L}, and another, $\mathcal{C}^{\text{R}}_{\text{aux}}$, to predict b_{R}.

Figure C.4. Samples from the Multi-Color MNIST dataset [17]. In the first row, images that are bias-aligned with respect to both background colors; in the second row, images that are bias-conflicting with respect to at least one of the colors.

CivilComments [3, 13] is a text dataset composed of comments posted in online discussion forums. The associated task usually corresponds to classifying whether a comment is toxic. Besides having a binary label y indicating toxicity, each comment is originally accompanied by binary labels $b_1, \ldots, b_8$ indicating the presence of a mention to each of the sensitive attributes *male*, *female*, *LGBT*, *black*, *white*, *Christian*, *Muslim*, *other religion* [9, 10]. We follow Izmailov *et al*. [10], Idrissi *et al*. [9] and Tiwari *et al*. [29], and consider the *coarse* version of the dataset, where the bias attributes are summarized into a single binary label b, which is equal to 1 if and only if at least one of the $b_1, \ldots, b_8$ is 1. Fig. C.5 shows some samples from the CivilComments dataset.

$(y, b) = (0, 0)$	$(y, b) = (0, 1)$
In my opinion...a good decision over a bad promise.	He sounds like a good man. I'm sorry for your loss.
$(y, b) = (1, 0)$	$(y, b) = (1, 1)$
Play stupid games, win stupid prizes	If only he were a Black woman we'd maybe put him on the dollar bill...

Figure C.5. Samples from the CivilComments dataset [3, 13]. Label y indicates whether a comment is toxic, while b indicates if it mentions any sensitive attribute.

D. Architectures and training settings

For BiasedMNIST, we employ the same architecture as in [1]: a fully convolutional network, composed of four layers of 7x7 kernels; each layer is followed by batch normalization and a ReLU activation. For training the vanilla model, the batch size is set to 100, and we use Stochastic Gradient Descent (SGD) with initial learning rate 0.1, momentum 0.9, and weight decay 10^{-4}, as in [28]; for the experiments with $\rho \in \{0.99, 0.995\}$, we train the vanilla model for 80 epochs, and the learning rate is divided by 10 at epochs 40 and 60. For $\rho \in \{0.997, 0.999\}$, the vanilla model is trained for 100 epochs, and the learning rate is divided by 10 at epochs 80 and 90. During BISE, a masking parameter m_i is assigned to each filter in the network.

For the experiments on CelebA and Corrupted-CIFAR10, we employ the ResNet18 architecture [6], pre-trained on ImageNet [26]. For CelebA, following [22], the vanilla model is trained for 50 epochs, using the Adam optimizer [12] with learning rate 10^{-4}, weight decay 10^{-4}, and remaining hyperparameters as PyTorch's [25] default values; the batch size is 256, and the training samples are augmented with random horizontal flip. For Corrupted-CIFAR10, the vanilla model is trained for 200 epochs, with batch size 256, and with the same optimizer, but with a learning rate initialized at 10^{-3} and reduced by cosine annealing, as in [2]; following [22], 32x32 random crop and random horizontal flip are applied to the training samples. When applying BISE, a masking parameter m_i is assigned to each ReLU-activated output in ResNet18 residual building blocks.

For Multi-Color MNIST, we follow [17]: we employ an MLP with three hidden layers, each of them with 100 hidden units, and, for the vanilla training, we use the Adam optimizer with learning rate 10^{-3} and weight decay 10^{-4}. We train the model for 100 epochs with batch size 256. During BISE, a mask parameter m_i is assigned to each neuron in the hidden layers.

For CivilComments, the vanilla model is obtained by following Izmailov *et al*. [10]: a BERT model (pre-trained on Book Corpus and English Wikipedia) [5] is trained on CivilComments for 10 epochs, using the AdamW optimizer [19], with learning rate initialized at 10^{-5} and linearly annealed, weight decay 10^{-4}, and batch size 16. For applying BISE, we assign mask parameters m_i to the neurons in the feed-forward layers and the pooler layer in BERT.

Finetuning settings. To finetune the BISE-extracted subnetwork, we employ the same optimizer that was used for training the vanilla model, with the original learning rate, except for BiasedMNIST and CivilComments, where the validation set is leveraged to select the learning rate: 10^{-3} for BiasedMNIST, Corrupted-CIFAR10 and Multi-Color MNIST, 10^{-4} for CelebA, and 10^{-8} for CivilComments. The finetuning is performed for 50 epochs, and, when a validation set is available, the best finetuned subnetwork is selected (according to the validation accuracy).

Experimental results reported for BISE (in the main paper and in the Supplementary Material) were obtained by averaging the results across three seeds. When computing the sparsity of the BISE-extracted subnetworks for BiasedMNIST, CelebA, Corrupted-CIFAR10 and Multi-Color MNIST, we have employed the *Simplify* library [4], with batch normalization fusion enabled.

E. On the number of parameters updated

Table E.1 shows the number of parameters m_i that are trained during BISE, against the number of weights present in the corresponding dense model, for each dataset. The number of mask parameters m_i is much smaller than the number of parameters in the network (since each m_i is associated with one neuron/filter, not with an individual weight).

Table E.1. Number of weights *vs.* of updated parameters m_i.

	Biased MNIST	**CelebA**	**Corrupted-CIFAR10**	**Multi-Color MNIST**	**Civil Comments**
# weights	531 210	11 177 538	11 181 642	256 510	109 361 664
# m_i	240 (0.05%)	3 840 (0.03%)	3 840 (0.03%)	300 (0.12%)	46 848 (0.04%)

F. Preliminary experiment

As a preliminary experiment, we compare BISE with another method proposed for identifying unbiased subnetworks, FFW [21]. Although the main objective of FFW is simply to showcase the existence of unbiased subnetworks, which are identified leveraging an *unbiased* training set (unrealistic setup), we believe it could be interesting to verify whether it could be applied in a case where the training set is *biased*. Considering the BiasedMNIST dataset, when a biased training set is used and given a vanilla-trained model with test accuracy $88.9_{\pm 0.4}\,\%$, FFW can extract a subnetwork that displays a test accuracy of $80.6_{\pm 6.7}\,\%$, while the one extracted through BISE showcases an accuracy of $96.1_{\pm 0.5}\,\%$ (see Tab. 1). Hence, FFW fails to debias the vanilla network when a biased training set is leveraged.

G. Results on BiasedMNIST for $\rho = 0.999$

In Tab. G.1, we present the results obtained for our method on the BiasedMNIST dataset with $\rho = 0.999$.

Firstly, we observe that, on the unbiased test set, the vanilla-trained model achieves accuracy corresponding to random guess (*i.e.*, 10%, as we have $C = 10$ classes). This phenomenon suggests that the vanilla model essentially relies only on the bias-related features (the background color) to predict the digits, without effectively learning the core, relevant features (*e.g.*, the digit shapes).

The subnetwork extracted with BISE achieves the accuracy of $15.7_{\pm 1.3}\,\%$. Although this is higher than the accuracy of the vanilla model, the proposed method could not isolate a well-performing subnetwork that would mostly rely on the features relevant to the task, without finetuning the remaining parameters. We also provide the results obtained with FFW [21]. The fact that FFW, a method that promotes debiasing by leveraging an *unbiased* training set, also fails to identify a well-performing unbiased subnetwork suggests that such a subnetwork may not exist in the case of severe level of spurious correlations in the training set $\mathcal{D}_{\text{train}}$ (as it is the case here with $\rho = 0.999$). However, we show that, by further finetuning the BISE-extracted subnetwork, we can considerably improve its performance. Effectively, as described in Sec. 4.5, if an unbiased substruc-

ture does not exist within the vanilla model, then BISE is not expected to extract a robust subnetwork, which impacts the general performance of our approach.

Table G.1. Results on BiasedMNIST for $\rho = 0.999$. (*) indicates that debiasing is performed by leveraging an unbiased dataset.

Method	**Acc. (%)** ↑	S **(%)** ↑	**MFLOPs** ↓
Vanilla	$10.0_{\pm 0.1}$	0	415.4
BISE	$15.7_{\pm 1.3}$	$85.6_{\pm 4.0}$	$59.8_{\pm 16.7}$
BISE + finetuning	$60.5_{\pm 9.4}$	$85.6_{\pm 4.0}$	$59.8_{\pm 16.7}$
BISE (last)	$14.7_{\pm 2.5}$	$78.2_{\pm 2.9}$	$90.5_{\pm 11.9}$
FFW * [21]	19.9	–	–

H. Sensitivity analysis on E, κ, υ and τ_{min}

In this section, we present a sensitivity study on the effect of the hyperparameters E (Tab. H.1), κ (Tab. H.2), υ (Tab. H.3) and τ_{min} (Tab. H.4) of the method proposed, on the BiasedMNIST dataset with $\rho = 0.99$. As described in Sec. 4.1, in our experiments we used $E = 50$ (the number of epochs for training/finetuning the auxiliary classifier $\mathcal{C}_{\text{aux}}$), $\kappa = 0.5$ (the factor by which τ is updated), $\upsilon = 10$ (the period of the τ updates) and $\tau_{\text{min}} = 10^{-3}$ (the minimum value of τ, which indicates when the algorithm should stop).

From the tables, we see that the proposed method demonstrates low sensitivity to variations in hyperparameters. In particular, Tab. H.1 indicates that, in the considered setup, there is no need to pre-train or finetune the auxiliary classifier $\mathcal{C}_{\text{aux}}$ for a large number of epochs. Additionally, Tables H.2 and H.3 suggest that the rate with which the temperature τ is reduced (determined by the factor κ and the period υ) does not significantly impact the extracted subnetwork. Finally, Tab. H.4 suggests that, after τ is reduced below 10^{-2}, the algorithm has already found a certain unbiased subnetwork, and executing the algorithm for longer does not lead to another, better-performing, subnetwork.

Table H.1. Effect of E on the extracted subnetwork.

Metric	E				
	1	5	10	20	50
Acc. (%) ↑	$95.9_{\pm 0.4}$	$96.0_{\pm 0.5}$	$96.0_{\pm 0.4}$	$95.7_{\pm 0.5}$	$96.1_{\pm 0.5}$
S **(%)** ↑	$18.9_{\pm 7.3}$	$17.6_{\pm 7.1}$	$20.0_{\pm 6.1}$	$17.2_{\pm 7.6}$	$20.9_{\pm 4.4}$
MFLOPs ↓	$336.7_{\pm 30.4}$	$342.1_{\pm 29.5}$	$331.9_{\pm 25.2}$	$343.7_{\pm 31.6}$	$328.3_{\pm 18.2}$

Table H.2. Effect of κ on the extracted subnetwork.

Metric	κ		
	0.1	0.5	0.8
Acc. (%) ↑	$96.0_{\pm 0.4}$	$96.1_{\pm 0.5}$	$96.2_{\pm 0.6}$
S **(%)** ↑	$20.6_{\pm 3.9}$	$20.9_{\pm 4.4}$	$18.9_{\pm 7.6}$
MFLOPs ↓	$329.3_{\pm 16.3}$	$328.3_{\pm 18.2}$	$336.5_{\pm 31.4}$

Table H.3. Effect of υ on the extracted subnetwork.

Metric	υ				
	1	5	10	20	50
Acc. (%) ↑	$95.9_{\pm 0.5}$	$95.8_{\pm 0.5}$	$96.1_{\pm 0.5}$	$96.0_{\pm 0.5}$	$96.5_{\pm 0.3}$
S **(%)** ↑	$22.4_{\pm 6.3}$	$18.3_{\pm 6.2}$	$20.9_{\pm 4.4}$	$19.2_{\pm 6.3}$	$19.8_{\pm 8.0}$
MFLOPs ↓	$322.2_{\pm 26.2}$	$338.9_{\pm 25.7}$	$328.3_{\pm 18.2}$	$335.2_{\pm 26.2}$	$333.0_{\pm 33.3}$

Table H.4. Effect of τ_{min} on the extracted subnetwork.

Metric	τ_{min}			
	10^{-1}	10^{-2}	10^{-3}	10^{-4}
Acc. (%) ↑	$95.9_{\pm 0.6}$	$96.1_{\pm 0.5}$	$96.1_{\pm 0.5}$	$96.1_{\pm 0.5}$
S **(%)** ↑	$20.4_{\pm 5.0}$	$20.9_{\pm 4.4}$	$20.9_{\pm 4.4}$	$20.9_{\pm 4.4}$
MFLOPs ↓	$330.3_{\pm 20.7}$	$328.3_{\pm 18.2}$	$328.3_{\pm 18.2}$	$328.3_{\pm 18.2}$

I. Updating batch normalization layers

We have conducted an experiment where, besides learning the pruning mask, we are also updating the parameters in the batch normalization layers. It has been shown, indeed, that in some tasks like domain adaptation just updating these few parameters can be sufficient to have a significant gain in performance [23]. In Tab. I.1, we show how learning the parameters in the batchnorm layers – specifically, updating the running statistics – can prospectively boost the performance. We did not include this procedure in the main approach as it would require an increment in the number of learned parameters. Table I.2 displays the sparsity and complexity of the extracted subnetworks.

Table I.1. Experiments on BiasedMNIST with trainable *vs.* non-trainable batch normalization layers.

Method	**Accuracy (%)**		
	$\rho = 0.99$	$\rho = 0.995$	$\rho = 0.997$
Vanilla	$88.9_{\pm 0.4}$	$75.1_{\pm 4.2}$	$66.1_{\pm 1.7}$
BISE (Tab. 1)	$96.1_{\pm 0.5}$	$92.2_{\pm 1.9}$	$90.8_{\pm 0.6}$
BISE + trainable BN (avg., std.)	$97.3_{\pm 0.1}$	$94.9_{\pm 1.1}$	$94.0_{\pm 1.2}$
BISE + trainable BN (avg., std., β, γ)	$96.9_{\pm 0.4}$	$92.2_{\pm 1.9}$	$90.1_{\pm 3.7}$

Table I.2. Sparsity (S) and computational cost (MFLOPs) of models, for the BiasedMNIST dataset, when updating the batch normalization layers during BISE.

Method	**Proportion of bias-aligned samples in the training set** (ρ)					
	$\rho = 0.99$		$\rho = 0.995$		$\rho = 0.997$	
	S (%) ↑	MFLOPs ↓	S (%) ↑	MFLOPs ↓	S (%) ↑	MFLOPs ↓
Vanilla	0	415.4	0	415.4	0	415.4
BISE (Tab. 2)	$20.9_{\pm 4.4}$	$328.3_{\pm 18.2}$	$29.9_{\pm 7.5}$	$290.8_{\pm 31.0}$	$35.0_{\pm 1.5}$	$269.6_{\pm 6.2}$
BISE + trainable BN (avg., std.)	$18.3_{\pm 6.8}$	$339.2_{\pm 28.1}$	$26.5_{\pm 4.0}$	$305.0_{\pm 16.7}$	$33.3_{\pm 7.9}$	$276.9_{\pm 32.9}$
BISE + trainable BN (avg., std., β, γ)	$32.7_{\pm 7.7}$	$279.5_{\pm 32.0}$	$45.7_{\pm 3.9}$	$225.6_{\pm 16.2}$	$48.2_{\pm 12.0}$	$214.9_{\pm 49.8}$

J. The auxiliary parameters $\{m_i\}$ provide a way to rank neurons

In this section, we show that the auxiliary variables $\{m_i\}$ represent a *ranking* of neurons/filters, indicating the priority with which we should prune them to accomplish the debiasing. Given a trained set $\{m_i\}$, let us modify the gating function as $\hat{h}_i = h_i \cdot \mathbf{1}\{\hat{m}_i \geq \zeta\}$, with $\zeta \in [0, 1]$ being the threshold for defining the boolean pruning mask, and considering $\hat{m}_i$ at $\tau = 1$, *i.e.*, $\hat{m}_i = \sigma(m_i)$. For the BiasedMNIST dataset with $\rho = 0.99$, Fig. J.1 shows the distribution of $\hat{m}_i$ in the debiased model, as well as how the sparsity and the performance on the unbiased test set vary as we modify ζ. In particular, no neurons are pruned for $\zeta = 0$, while all the neurons are pruned for $\zeta = 1$. As we observe, for $\zeta = 0$, the accuracy corresponds to the performance of the vanilla model, and, as we increase ζ, we achieve a maximum performance around $\zeta = 0.5$ (which consists of the threshold originally used during the training of $\{m_i\}$). In the sequence, as ζ approaches 1, the accuracy progressively drops to random guess (10%), as expected, since we perform pruning without finetuning the remaining network parameters. We observe that the steepest drop in accuracy and increase in sparsity happen when ζ is around the mode of the distribution of $\hat{m}_i$, due to the removal of a large number of parameters.

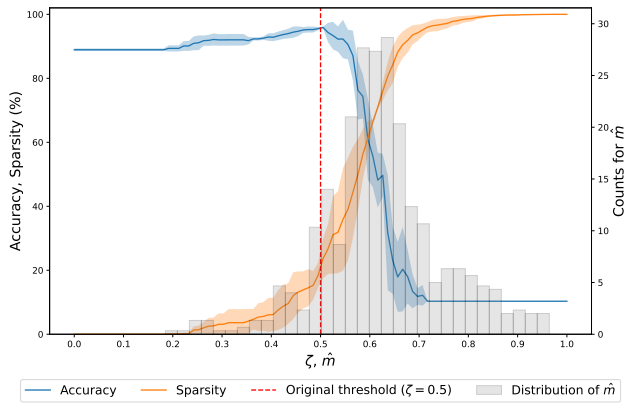

Figure J.1. Analysis of the effect of varying the threshold ζ on the pruned network. Threshold $\zeta = 0$ corresponds to the vanilla dense model; $\zeta = 0.5$ is the original threshold used in BISE.

K. Unsupervised debiasing on BiasedMNIST

In Tab. K.1 and Tab. K.2, we present the results for BISE on BiasedMNIST, in the *unsupervised* debiasing scenario. The identification model, $\tilde{f}$, is trained for only one epoch. The predictions $\tilde{b}$ from $\tilde{f}$ closely reflect the true color b, hence making the unsupervised procedure close to the supervised debiasing case. Our results are generally competitive with other unsupervised debiasing approaches, such as LfF [22] and SoftCon [8] (*cf*. Tab. 1 in the main paper).

Table K.1. Unsupervised (*Unsup.*) debiasing on BiasedMNIST.

Method	**Accuracy (%)**		
	$\rho = 0.99$	$\rho = 0.995$	$\rho = 0.997$
Vanilla	$88.9_{\pm 0.4}$	$75.1_{\pm 4.2}$	$66.1_{\pm 1.7}$
Unsup. BISE	$95.8_{\pm 0.5}$	$92.0_{\pm 2.0}$	$90.4_{\pm 0.6}$
Unsup. BISE (last)	$95.4_{\pm 0.7}$	$90.6_{\pm 1.9}$	$90.0_{\pm 0.2}$

Table K.2. Sparsity (S) and computational cost (MFLOPs) of models, for the BiasedMNIST dataset, in the unsupervised debiasing scenario.

Method	**Proportion of bias-aligned samples in the training set (ρ)**					
	$\rho = 0.99$		$\rho = 0.995$		$\rho = 0.997$	
	S (%) ↑	MFLOPs ↓	S (%) ↑	MFLOPs ↓	S (%) ↑	MFLOPs ↓
Vanilla	0	415.4	0	415.4	0	415.4
Unsup. BISE	$19.9_{\pm 4.4}$	$332.4_{\pm 18.2}$	$26.2_{\pm 4.7}$	$306.3_{\pm 19.5}$	$30.9_{\pm 2.3}$	$286.6_{\pm 9.7}$
Unsup. BISE (last)	$19.7_{\pm 3.6}$	$333.4_{\pm 14.9}$	$28.8_{\pm 4.9}$	$295.6_{\pm 20.3}$	$28.8_{\pm 4.5}$	$295.5_{\pm 18.7}$

L. Robustness to noisy labels

To complement the study from Appendix K, we have conducted experiments on BiasedMNIST, with $\rho = 0.99$, to show how BISE is robust to *noisy* bias labels – which can be usual in practical applications. We proceed as follows: before applying BISE, we randomly select a fraction $p \in [0, 1]$ of the training samples, and change their bias label b to another random value. The results for test accuracy are reported in Fig. L.1. We observe that, for all noise levels considered, BISE could improve the test accuracy, in comparison to the vanilla model, even if by a small margin. Importantly, even under high noise, our method does not extract a subnetwork that is worse than the vanilla model.

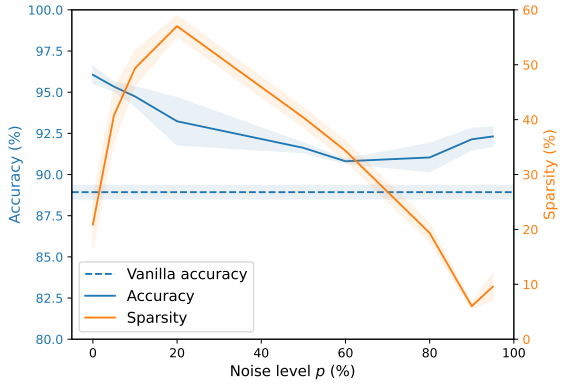

Figure L.1. Results on BiasedMNIST (training set with $\rho = 0.99$) under noisy bias labels. Noise level $p = 0\%$ corresponds to the standard setting (*i.e.*, use of true bias labels).

M. On the pruning of bias-related features

As a simple empirical assessment of whether the pruned parameters correspond to bias-relevant features, we have conducted the following experiment. We consider the BiasedMNIST setup and the two following models:

- a vanilla model, trained on a biased set with $\rho = 0.99$;
- the corresponding debiased subnetwork, obtained with BISE.

For each model, we attach an auxiliary classifier to the output of the corresponding encoder (*i.e.*, right before the last layer); the auxiliary classifiers are trained to predict the color of the digits, b, based on the latent representation from the respective encoder's output. In this specific study, we perform this training using an *unbiased* version of the BiasedMNIST training set (*i.e.*, with $\rho = 0.1$), to avoid the auxiliary classifiers from leveraging digit-related features (due to the correlation digit-color in a biased set). The training is performed for 100 epochs, using SGD with learning rate 0.1, weight decay 10^{-4}, momentum 0.9. The results on color prediction by the auxiliary classifiers are reported in Tab. M.1.

Table M.1. Comparison of accuracy on color prediction for two auxiliary classifiers: one trained on top of a vanilla encoder, and the other trained on top of the BISE-pruned encoder.

Model	**Accuracy of the aux. classifier on color prediction (%)**
Vanilla	$97.6_{\pm 1.0}$
Debiased with BISE	$82.1_{\pm 7.3}$

We observe that, although the bias-related features (*i.e.*, related to the color) are not entirely removed from the latent representation provided by the BISE-pruned encoder, in this case, the color prediction is made "harder", as indicated by the significantly lower color prediction accuracy. This observation suggests that, by applying our method to prune the network, we can reduce the impact of bias on the information that can be leveraged by a classifier attached to the encoder. In other words, it is suggested that, by applying BISE, the representation $\hat{z}$ provided by the pruned encoder becomes more invariant to the bias, hence making it harder to predict b from $\hat{z}$.

N. Summary table

In Tab. N.1, we present a summary of the results across the different datasets considered. The competing methods correspond to the ones presented in the tables from Sec. 4.2.

Metric	Model	Dataset				
		BiasedMNIST ($\rho = 0.99$)	**Corrupted-CIFAR10** ($\rho = 0.95$)	**CelebA**	**Multi-Color MNIST** ($\rho_{\mathrm{L}} = 0.99$, $\rho_{\mathrm{R}} = 0.95$)	**CivilComments**
Acc. (%)	Vanilla	$88.9_{\pm 0.4}$	$47.18_{\pm 0.34}$	$76.5_{\pm 2.1}$	$58.2_{\pm 0.6}$	$59.6_{\pm 2.7}$
	Best competitor	98.1 (BCon+BBal [8])	51.13 (DFA [16])	91.4 (BCon+BBal [8])	73.1 (VCBA [20])	80.4 (Group DRO [27])
	BISE*	$96.1_{\pm 0.5}$	$55.38_{\pm 1.96}$	$89.7_{\pm 0.8}$	$60.3_{\pm 1.0}$	$80.4_{\pm 0.2}$
	BISE + finetuning*	$98.1_{\pm 0.1}$	$57.22_{\pm 1.81}$	$91.8_{\pm 1.3}$	$70.6_{\pm 1.6}$	$81.0_{\pm 0.1}$
S **(%)**	BISE*	$20.9_{\pm 4.4}$	$82.3_{\pm 0.6}$	$67.6_{\pm 0.8}$	$17.1_{\pm 5.3}$	$26.0_{\pm 5.4}$
FLOPs	Vanilla/competitors	415.4 M	37.1 M	1818.6 M	256.2 k	–
	BISE*	$328.3_{\pm 18.2}$ M	$22.5_{\pm 0.6}$ M	$821.5_{\pm 33.1}$ M	$212.3_{\pm 13.5}$ k	–

Table N.1. Summary of results across datasets. (*) indicates that the result for *BISE "best"* is reported whenever a validation set is available; otherwise, we report the result for *BISE "last"*. The metric *Acc.* corresponds to the accuracy on an unbiased test set, in the case of BiasedMNIST, CelebA and Corrupted-CIFAR10; for Multi-Color MNIST, it denotes the average test accuracy across the four existing groups in the dataset (as described in Sec. 4.2); for CivilComments, it denotes the worst-group test accuracy. The sparsity (S) for the vanilla and competing methods is zero, since no parameter is removed from the network.

References

[1] Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. Learning de-biased representations with biased representations. In *ICML*, 2020. 1, 2, 3

[2] Carlo Alberto Barbano, Benoit Dufumier, Enzo Tartaglione, Marco Grangetto, and Pietro Gori. Unbiased supervised contrastive learning. In *ICLR*, 2023. 4

[3] Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion proceedings of the 2019 world wide web conference*, 2019. 3

[4] Andrea Bragagnolo and Carlo Alberto Barbano. Simplify: A python library for optimizing pruned neural networks. *SoftwareX*, 2022. 4

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019. 4

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4

[7] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019. 1, 3

[8] Youngkyu Hong and Eunho Yang. Unbiased classification through bias-contrastive and bias-balanced learning. In *NeurIPS*, 2021. 1, 6, 8

[9] Badr Youbi Idrissi, Martin Arjovsky, Mohammad Pezeshki, and David Lopez-Paz. Simple data balancing achieves competitive worst-group-accuracy. In *Proceedings of the First Conference on Causal Learning and Reasoning*. PMLR, 2022. 3

[10] Pavel Izmailov, Polina Kirichenko, Nate Gruver, and Andrew G Wilson. On feature learning in the presence of spurious correlations. In *NeurIPS*, 2022. 3, 4

[11] Nayeong Kim, Juwon Kang, Sungsoo Ahn, Jungseul Ok, and Suha Kwak. Improving robustness to multiple spurious correlations by multi-objective optimization. In *ICML*, 2024. 1, 2, 3

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 4

[13] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton Earnshaw, Imran Haque, Sara M Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. Wilds: A benchmark of in-the-wild distribution shifts. In *ICML*, 2021. 3

[14] Alex Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, Department of Computer Science, University of Toronto, 2009. 3

[15] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998. 2

[16] Jungsoo Lee, Eungyeup Kim, Juyoung Lee, Jihyeon Lee, and Jaegul Choo. Learning debiased representation via disentangled feature augmentation. In *NeurIPS*, 2021. 8

[17] Zhiheng Li, Anthony Hoogs, and Chenliang Xu. Discover and mitigate unknown biases with debiasing alternate networks. In *ECCV*, 2022. 2, 3, 4

[18] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 1, 2, 3

[19] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 4

[20] Rémi Nahon, Van-Tam Nguyen, and Enzo Tartaglione. Mining bias-target alignment from voronoi cells. In *ICCV*, 2023. 8

[21] Rémi Nahon, Ivan Luiz De Moura Matos, Van-Tam Nguyen, and Enzo Tartaglione. Debiasing surgeon: fantastic weights and how to find them. In *ECCV*, 2024. 2, 4, 5

[22] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: De-biasing classifier from biased classifier. In *NeurIPS*, 2020. 3, 4, 6

[23] Leonardo Olivi, Edoardo Santero Mormile, and Enzo Tartaglione. Efficient adaptation of deep neural networks for semantic segmentation in space applications. *Scientific Reports*, 2025. 5

[24] Geon Yeong Park, Chanyong Jung, Sangmin Lee, Jong Chul Ye, and Sang Wan Lee. Self-supervised debiasing using low rank regularization. In *CVPR*, 2024. 1

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 4

[26] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015. 4

[27] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *ICLR*, 2020. 1, 2, 3, 8

[28] Enzo Tartaglione. Information removal at the bottleneck in deep neural networks. *BMVC*, 2022. 3

[29] Rishabh Tiwari, Durga Sivasubramanian, Anmol Mekala, Ganesh Ramakrishnan, and Pradeep Shenoy. Using early readouts to mediate featural bias in distillation. In *WACV*, 2024. 3

[30] Christos Tsirigotis, Joao Monteiro, Pau Rodriguez, David Vazquez, and Aaron C. Courville. Group robust classification without any group information. In *NeurIPS*, 2023. 1, 3