FROM BIASED DATA TO UNBIASED MODELS: A META-LEARNING APPROACH

Anonymous authors
Paper under double-blind review

ABSTRACT

It is well known that large deep architectures are powerful models when adequately trained, but may exhibit undesirable behavior leading to confident incorrect predictions, even when evaluated on slightly different test examples. Test data characterized by distribution shifts (from training data distribution), outliers, and adversarial samples are among the types of data affected by this problem. This situation worsens whenever data are biased, meaning that predictions are mostly based on spurious correlations present in the data. Unfortunately, since such correlations occur in the most of data, a model is prevented from correctly generalizing the considered classes. In this work, we tackle this problem from a meta-learning perspective. Considering the dataset as composed of unknown biased and unbiased samples, we first identify these two subsets by a pseudo-labeling algorithm, even if coarsely. Subsequently, we apply a bi-level optimization algorithm in which, in the inner loop, we look for the best parameters guiding the training of the two subsets, while in the outer loop, we train the final model taking benefit from augmented data generated using Mixup. Properly tuning the contributions of biased and unbiased data, followed by the regularization introduced by the mixed data has proved to be an effective training strategy to learn unbiased models, which show superior generalization capabilities. Experimental results on synthetically and realistically biased datasets surpass state-of-the-art performance, as compared to existing methods.

1 INTRODUCTION

In classification tasks, it is widely recognized that deep learning architectures can learn large amount of data, reaching unprecedented outstanding performance. However, such models are also very sensitive to data, meaning that they are prone to errors with high confidence whenever test samples are drawn from a distribution different from that of the training set. One reason is that, in certain conditions, these models have problems to generalize well the classes considered as they likely memorize the training data rather than learning the salient characteristics of each category of examples. This behavior is especially evident when training data are biased, i.e., samples include spurious correlations with class labels or, in other words, the trained model learns some "shortcuts" to classify data, so failing to generalize the class properly. For example, a fish can be classified as such due to the presence of the blue sea and not for the fish semantic information, hence a model likely fails in case the input image depicts a fish located in a brown table market. Such shortcuts are learnt since most of the samples are characterized by the bias (fishes in the sea) while only a few samples are unbiased (fishes in unusual contexts), which prevents from generalizing the class properly.

When optimizing models under the presence of biased data, the ground-truth knowledge of the bias is typically beneficial. For instance, having an additional annotation regarding whether the fish is in the sea or not can be used to drive the optimization towards a data representation invariant to such attribute (See Figure 1(a)). Several methods approached the problem in this way and sought for a data representation invariant to a known factor (Alvi et al. (2018); Kim et al. (2019); Li & Vasconcelos (2019); Wang et al. (2019); Ragonesi et al. (2020); Sagawa et al. (2019)): we term this problem *supervised debiasing*, i.e. the knowledge of the bias acts as an auxiliary data annotation that can be useful to consider in training in order to get invariance with respect to it.

However, the hypothesis of having an additional label is unrealistic in most practical scenarios as it requires great effort during data annotation, and in some cases can even be impossible whenever the

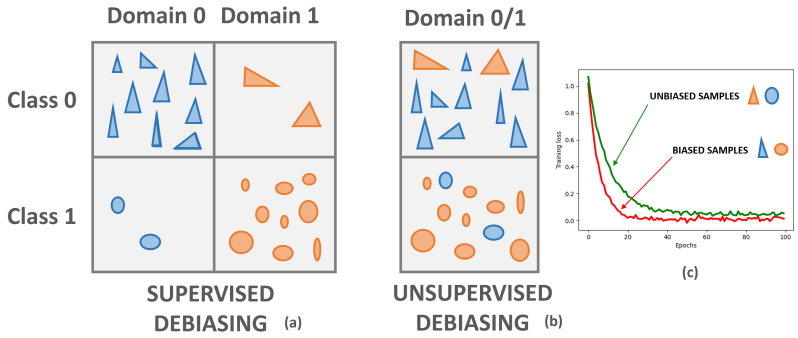


Figure 1: **Problem description.** (a) Biased dataset occurs when there is an imbalance regime regarding pairs (class, domain), where each class is observed mostly under one distribution, leaving other options under-represented. This results in trained models which do not generalize well. In the case of supervised debiasing case, one has additional annotations regarding the domain distribution. (b) In the unsupervised debiasing case, one has only access to the class labels. A possible approach to distinguish biased/unbiased samples is via pseudo-labeling. (c) The plots show that the loss for the biased samples are decreasing much faster than the loss for unbiased samples, proving that the former can be learnt more easily than the latter. (Best viewed in color)

control of data gathering is unfeasible, hence the urge of methods that can generalize even without this additional supervision.

For these reasons, we face here the more challenging setting of the *unsupervised debiasing* problem, i.e., we assume that the ground-truth knowledge of the bias is not readily available. Hence, we attempt to (implicitly) infer this information while debiasing our model and achieving a successful generalization on the test set (See Figure 1(b)).

In this paper, we devised a two-stage algorithm tackling the unsupervised debiasing problem. First, we separate biased from unbiased samples through a pseudo-labeling approach. Second, equipped with such (noisy) pseudo-labels, we manage the problem of learning from this data using a Meta-Learning approach (inspired by Finn et al. (2017)) to produce a data representation that can accommodate both biased and unbiased samples. The method consists of a bi-level optimization strategy, in which learning from biased and unbiased samples are treated as meta-tasks in the inner loop, while the outer loop uses augmented samples as a meta-validation task. We use MixUp (Nam et al. (2020)) as the technique to augment the dataset producing meta-validation data: in this way, we feed the model with data that can be as much "neutral" as possible by mixing samples of the biased split with those of the unbiased split. We aim to produce synthetic samples which are unusual, and therefore represents cases that are under-represented in the original training data, overall regularizing and improving the training. We show that mixing the two subsets brings improvement not only for the biased samples but also keeps high accuracy on the biased ones, avoiding catastrophic forgetting.

We validate our method on several benchmarks that are both synthetic with controlled bias (colored MNIST and Corrupted CIFAR-10) and more realistic (Waterbirds and BAR), showing outstanding performance as compared with existing methods.

To recap, the contributions of our work are:

- We propose to face the unsupervised debiasing problem by introducing a two-stage approach that, after the initial coarse identification of the biased and unbiased samples, can modulate the contribution of each example during the model training by a meta-learning strategy.
- Specifically, we consider learning from biased and unbiased samples as separate meta-tasks, and we *generate* new data by augmentation, which we treat as a (meta-)validation task. By jointly optimizing the original meta-training tasks and the generated meta-validation task, we inject a strong regularization in the training process leading to more general representation learning.
- Our approach, validated on datasets with controlled bias and realistic benchmarks, showed to outperform state-of-the-art performance by a significant margin.

The rest of the paper is organized as follows. In Section 2, we describe the works close to our proposal, highlighting the original aspects introduced. Section 3 reports our method, where we detail our two-stage approach. Section 4 presents the results and a thorough ablation analysis. Section 5 wrap-ups the work and sketches the future research directions.

2 RELATED WORK

Learning from biased data can be seen as a specific case of Out-Of-Distribution (OOD) domain generalization. This topic has been addressed with different methodologies, including meta-learning. Here, we briefly review the most related literature.

Learning from biased data. The problem of learning from biased data has been explored in past years in the supervised debiasing setting, i.e. when labels for the factor (bias) to be removed are readily available. Several methods approached the problem seeking an invariant data representation to a known factor. Such approaches rely on adversarial learning (Alvi et al. (2018); Kim et al. (2019)), variational inference (Moyer et al. (2018)), Information Theory (Ragonesi et al. (2020)), re-sampling strategies (Li & Vasconcelos (2019)), or robust optimization (Sagawa et al. (2019).

Few recent works (Bahng et al. (2020); Levy et al. (2020); Nam et al. (2020); Liu et al. (2021)) have addressed the unsupervised debiasing problem that we face in this work. Bahng et al. (2020) formalizes the *cross-bias* problem where malicious shortcuts exist, easing the fit of training data, whereas the same shortcuts result useless for the inference stage. This hampers the model's generalization capability: the solution is learning a debiased model which is statistically independent from the one computed by a parallel computational stream that is guaranteed to be affected by the bias by design. In Nam et al. (2020), the nature of the aforementioned "shortcuts" are analyzed in terms of fitting speed at training time. Nam et al. show that biased samples are learnt faster than the unbiased ones. The relative difficulty of each sample is cast into a weight that modulates its learning rate: in this way, at training time, it is given more importance to the few outlying samples that do not follow the shortcuts. To this end, an ensemble of networks is trained, similarly to Bahng et al. (2020). Levy et al. (2020) provide statistical bounds and tackle the problem via robust optimization, considering a worst case loss of a sub-population of the dataset (typically samples with the highest loss). In Liu et al. (2021), the training data is split into two subsets relying on the predictions of a baseline model. The most difficult samples (likely those that do not follow shortcuts), are then upsampled.

Our work does not rely on an ensemble of networks to have a reference biased model. Instead, we perform a pseudo-labeling approach to split the dataset in two subsets and then treat them as two separate tasks to be learned via meta-learning. We also avoid data upsampling as in Liu et al. (2021) and Li & Vasconcelos (2019), whereas we pursue a data augmentation approach to combine biased and unbiased samples. Inspired by Mixup (Zhang et al. (2017)), we mix factors which are peculiar of the bias regime (likely representing a shortcut to infer the class) with those that do not follow such rules. The newly generated samples are expected to break the spurious correlations that affect the original data and allow the model to better generalize.

Meta-Learning for Out-Of-Distribution domain generalization. A class of meta-learning methods based on bi-level optimization (e.g., Model Agnostic Meta-Learning (Finn et al. (2017))), relies on an inner-loop stage optimizing model's meta-parameters on source data, and an outer-loop stage that updates the model parameters on (meta-)validation data. This nested optimization which involves computing a gradient through a gradient, has been shown to be effective for a fast adaptation of the model to the validation data. The goal is learning from an (empirical) training task distribution so to generalize and learn faster (i.e., with fewer samples) the validation task.

Subsequently, other methods have tackled the problem of Domain generalization (DG) (Li et al. (2017); Balaji et al. (2018); Li et al. (2019), to cite a few), casting the problem of learning from multiple tasks to learning from multiple distributions/domains. We adopt the same general scheme, however we face a considerably distinct problem: while in DG, different domains are fairly balanced, we deal with a severe data imbalance, that is, biased vs. unbiased, seen here as domains. This domain data imbalance is so dramatic that the model likely learns domain attributes to perform inference, hampering its generalization capabilities. This requires a tailored solution that we found effective through data augmentation, in order to attempt to reduce the imbalance problem. Moreover, differently from previous methods that rely on multiple source domains, we relax the hypothesis of having domain labels. Hence, we apply a pseudo-labeling method to discriminate the training set in two subsets that corresponds to different distributions. In fact, since one needs a meta-validation set to train the outer loop, our solution is to produce it, by generating synthetic validation data using data augmentation. This resulted quite effective even if the two subsets are noisy, that is, even if they do not perfectly identify the real distributions.

3 THE METHOD

We consider supervised classification problems with a training set $\mathcal{D}_{train} = \{x_k, y_k, d_k\}_{k=1}^N$, where x_k are raw input data, y_k class labels and d_k domain labels. In the case of a biased dataset, $\mathcal{D}_{train}$ has several classes $y^i, i = 1, ..., C$, which are considered to be observed under different domains $d^j, j = 1, ..., D$; D can be different from C but here, for clarity and without losing generality, we consider the case of $D = C$. When the majority of samples of a specific class y^i is observed under a single domain d^j, while other domains are under represented in the dataset, we say that the pair (y^i, d^j) is biased, i.e. there is a spurious correlation between class and domain.

We define $\mathcal{D}_{bias}$ as the subset of training samples that exhibit spurious correlations and $\mathcal{D}_{unbias}$ as the subset of samples with under represented pairs. Such subsets are highly imbalanced, i.e. $|\mathcal{D}_{bias}| >> |\mathcal{D}_{unbias}|$. For instance, in a cats vs. dogs classification problem, most of the cats may be observed in an indoor home environment, while most of the dogs may be observed in outdoor scenes. For both classes, very few images are outside of the main distribution.

We aim to tackle the *unsupervised debiasing* problem, which means that we do not have access to domain labels d nor to other bias information, hence we can just consider a training set only containing input data and class label, $\mathcal{D} = \{x_k, y_k\}_{k=1}^N$.

We want to train a parametric inference model $p_\theta(\mathbf{y}|\mathbf{x})$ on $\mathcal{D}$ to be deployed on test data $\mathcal{D}_{test}$ not seen during training. A neural network f_θ, with parameters θ, is used to approximate the distribution $p_\theta(\mathbf{y}|\mathbf{x})$. The parameters θ are usually found via Empirical Risk Minimization (ERM), i.e. minimizing the expected Cross-Entropy loss over the training data:

$$\theta^* = \arg\min_\theta \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{y}\sim\mathcal{D}} \mathcal{L}(\mathcal{D}, f_\theta), \text{ where } \mathcal{L}(\mathcal{D}, f_\theta) = \mathbf{y}^T \log(\sigma(f_\theta(\mathbf{x}))), \tag{1}$$

where σ is the softmax function. In such scenario, when trained via ERM, a model focuses mostly on the more numerous biased samples, underfitting the unbiased ones: this results in a biased model that uses spurious correlation (e.g., background) as a possible way to make inference, instead of correctly learning the class semantic. In general, $\mathcal{D}_{test}$ follows a data distribution different from $\mathcal{D}_{train}$, i.e. the biased pairs may be not the majority of samples. Hence it is important to have a model that can be deployed on both biased and unbiased pairs.

Our method tackles the unsupervised debiasing problem with a two-stage approach. In the first stage, we separate biased from unbiased samples through a pseudo-labeling algorithm. Equipped with such pseudo-labels, we train a model to produce a data representation that can accommodate both biased and unbiased samples. In the following, we detail the two main stages of our method.

3.1 BIAS IDENTIFICATION

In this stage, our goal is to split the training set $\mathcal{D}$ into two disjoint subsets $\hat{\mathcal{D}}_{bias}$ and $\hat{\mathcal{D}}_{unbias}$ that should resemble the actual, ground-truth $\mathcal{D}_{bias}$ and $\mathcal{D}_{unbias}$. In Nam et al. (2020), it is shown how the biased samples are learnt faster than the unbiased ones: the imbalanced nature of the dataset makes the model more prone to learn first the numerous biased samples and later those unbiased. This behaviour can be observed by looking at the loss function trends of the two subsets (See Fig. 1(c)). We exploit the fact that samples from D_{bias} are easily learnt during training, to design a strategy for splitting the dataset. We train a neural network f_ϕ via ERM until it reaches a training accuracy of γ, where γ is a hyper-parameter denoting the target accuracy. When the model reaches the desired accuracy level, the training stops and a forward pass of the entire training set is performed. Now, samples that are correctly predicted are assigned to $\hat{\mathcal{D}}_{bias}$ while those not correctly predicted are assigned to $\hat{\mathcal{D}}_{unbias}$. More formally:

$$\begin{aligned} \hat{\mathcal{D}}^\gamma_{bias} &= \{x \in \mathcal{D} \mid \sigma(f^\gamma_\phi(x)) = y\} \\ \hat{\mathcal{D}}^\gamma_{unbias} &= \{x \in \mathcal{D} \mid \sigma(f^\gamma_\phi(x)) \neq y\} \end{aligned} \tag{2}$$

Using γ as hyper-parameter is convenient for two reasons. First, our setting of the amount of desired accuracy is dataset agnostic. This is different from prior work (Liu et al. (2021)) that employs a similar strategy, but with the hyper-parameter controlling the number of epochs to train the model: in that case, the number of epochs are strictly dependent on the dataset that the model is trained

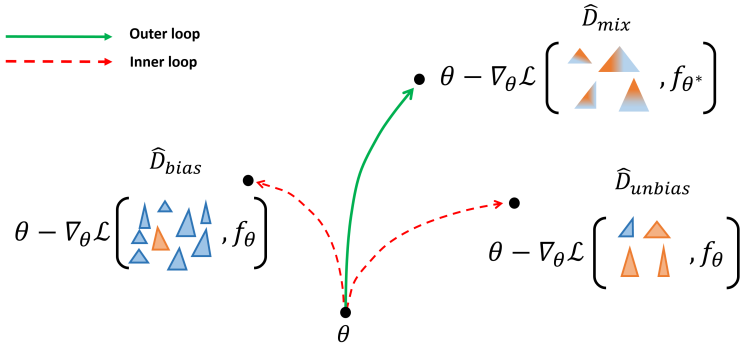


Figure 2: Starting from the current parameter configuration θ, gradients on $\mathcal{L}(\hat{\mathcal{D}}_{bias}, f_\theta)$ and $\mathcal{L}(\hat{\mathcal{D}}_{unbias}, f_\theta)$ are evaluated to produce the new configuration θ^*. The regularization step using mixed data aims at producing a contribution that decreases the loss function on $\hat{\mathcal{D}}_{bias}$, $\hat{\mathcal{D}}_{unbias}$, and $\hat{\mathcal{D}}_{mix}$, simultaneously, the latter estimated over the configuration θ^*. (Best viewed in color)

on. Second, we can have a precise control of the amount of samples assigned to the two splits, e.g. $\gamma = 0.85$ implies that 85% of training data are assigned to $\hat{\mathcal{D}}_{bias}$ and 15% to $\hat{\mathcal{D}}_{unbias}$. Obviously, especially in real use cases, we do not know the correct assignments of the samples to the splits, so we have to rely on a priori setting of this parameter.

3.2 Bias-invariant representation learning

Provided with pseudo-labels for the two estimated subsets $\hat{\mathcal{D}}_{bias}$ and $\hat{\mathcal{D}}_{unbias}$, we deal with the problem of learning data representations that are not only good for the biased data but can generalize well to unbiased samples. We adopt a neural network f_θ, trained from scratch, and we designed a bi-level optimization algorithm inspired by meta-learning to learn efficiently from such data.

Inner loop step. This is a meta-training step where we seek the best parameters θ for the two subsets $\hat{\mathcal{D}}_{bias}$ and $\hat{\mathcal{D}}_{unbias}$ via gradient descent:

$$\theta^* = \theta - \eta \, \nabla_\theta \left[(1-\gamma) \, \mathcal{L}(\hat{\mathcal{D}}_{bias}, f_\theta) + \gamma \, \mathcal{L}(\hat{\mathcal{D}}_{unbias}, f_\theta) \right] \tag{3}$$

where η is the learning rate. In this step, the two splits of the training data are treated as two separate tasks: we scale the two loss functions with two coefficients to deal with data imbalance ($|\hat{\mathcal{D}}_{bias}| >> |\hat{\mathcal{D}}_{unbias}|$). To rebalance the contributions from the two splits, an obvious choice is to set weights inversely proportional to the cardinality of the two subsets, which is nothing else than the fixed and controllable hyper-parameter γ.

Outer loop step. Standard meta-learning usually optimizes for the meta-test task using the parameters found in the inner loop, relying on a (typically small and clean) validation set. Here, we get rid of this assumption since do not have access to any held-out nor clean data, therefore we opt for a data augmentation approach in order to provide unseen data to the model.

We seek a representation that can conciliate both biased and unbiased samples and at the same time prevent the model from overfitting the meta-training data (the two subsets $\hat{\mathcal{D}}_{bias}$ and $\hat{\mathcal{D}}_{unbias}$), which is a common problem in meta-learning. We take inspiration from Mixup (Zhang et al. (2017)) as a way to combine samples from the two subsets. Mixup provides a convex combination of both input samples and labels and it has demonstrated its efficacy as an effective regularizer. Specifically, we feed the model with samples resulting from the mix of examples from biased and unbiased data, aiming at likely breaking the shortcuts present in the dataset (see Fig. 2).

We construct $\hat{\mathcal{D}}_{mix}$ by mixing samples of $\hat{\mathcal{D}}_{bias}$, $\hat{\mathcal{D}}_{unbias}$, sampling the parameter $\lambda \sim Beta(\alpha, \beta)$:

$$\begin{gathered} x_{mix} = \lambda \, \hat{x}_1 + (1-\lambda) \, \hat{x}_2 \\ y_{mix} = \lambda \, \hat{y}_1 + (1-\lambda) \, \hat{y}_2 \\ \hat{x}_1, \hat{y}_1 \in \mathcal{D}_{bias} \, , \hat{x}_2, \hat{y}_2 \in \mathcal{D}_{unbias} \end{gathered} \tag{4}$$

Computed the augmented samples x_{mix}, y_{mix}, the model is updated in the outer loop:

$$\mathcal{L} := \underbrace{(1-\gamma) \, \mathcal{L}(\hat{\mathcal{D}}_{bias}, f_\theta) + \gamma \, \mathcal{L}(\hat{\mathcal{D}}_{unbias}, f_\theta)}_{\text{Weighted ERM}} + \zeta \underbrace{\mathcal{L}(\hat{\mathcal{D}}_{mix}, f_{\theta^*})}_{\text{Regularizer}} \tag{5}$$

Algorithm 1 Learning to learn unbiased representations

Input: Dataset $\mathcal{D}$, initialized weights θ_0, learning rate η, hyper-parameters ζ, γ, T.
Output: learned weights θ
Initialize: $\theta \leftarrow \theta_0$
Identify $\hat{\mathcal{D}}_{bias}$ and $\hat{\mathcal{D}}_{unbias}$ by a pseudo-labeling method (Eq. 2) (γ controls the accuracy)
for t $= 1, ..., T$ **do**
    Sample $(\mathbf{x_b}, \mathbf{y_b})$, $(\mathbf{x_u}, \mathbf{y_u})$ uniformly from $\hat{\mathcal{D}}_{bias}$ and $\hat{\mathcal{D}}_{unbias}$
    Compute θ^* (Eq. 3) ▷ Inner loop step
    Sample $(\hat{\mathbf{x}}_\mathbf{1}, \hat{\mathbf{y}}_\mathbf{1})$, $(\hat{\mathbf{x}}_\mathbf{2}, \hat{\mathbf{y}}_\mathbf{2})$ uniformly from $\hat{\mathcal{D}}_{bias}$ and $\hat{\mathcal{D}}_{unbias}$
    Construct $\hat{\mathcal{D}}_{mix}$ (Eq. 4) ▷ Produce augmented samples
    Update θ (Eq. 5) ▷ Outer loop step

where ζ is a hyper-parameter controlling the regularization. Note that the first two losses are evaluated on the current parameters configuration θ, while the loss over the augmented samples is evaluated in the meta-state θ^* (see Eq. 3). This implies that the model has to compute a gradient through a gradient, similarly to what happens in optimization-based meta-learning methods. The hyperparameter ζ controls the amount of regularization in the final loss: if $\zeta = 0$, the method corresponds to a (weighted) ERM in which the contributions of the losses on the two subsets are scaled by $(1 - \gamma)$ and γ. When $\zeta > 0$ the weighted ERM optimization trajectory is corrected by the regularization term. This corresponds to find parameters θ that are good for both $\hat{\mathcal{D}}_{bias}$ and $\hat{\mathcal{D}}_{unbias}$, but can also possibly reduce the loss value on the newly generated data samples $\hat{\mathcal{D}}_{mix}$. Accuracy is not so affected by the choice of the ζ value: indeed, it increases as long as this ζ assumes positive values up to reaching high performance quite steadily, after that the contribution of the regularization becomes too strong and accuracy decreases. We set the value of ζ to a fixed value ($= 10$) for all experiments. Further analysis on this parameter is reported in the Appendix. The complete method is summarized in Algorithm 1.

4 Experiments

In the following, we show the effectiveness of models trained by our method in a series of benchmarks, ranging from toy problems with synthetic biases to realistic image classification applications. We compare with methods that tackle the same bias problem in both supervised and unsupervised way.

4.1 Synthetic bias: Colored MNIST and Corrupted CIFAR-10

To control the bias in the data and for the sake of comparison, we adopt two benchmarks that have been employed by Nam et al. (2020), namely colored MNIST and corrupted CIFAR-10[1,2]. The first is a modified version of the standard digit recognition dataset (LeCun & Cortes (2010)), in which colors are added in order to artificially induce a bias in the dataset. The dataset is made of 60,000 training RGB images and 10 classes to be predicted. Specifically, each sample is colored with a color tone which is randomly sampled from a Gaussian distribution whose mean is specific for each class; in practice, each class in the training data is observed mostly under a certain color tone, while the test set has no specific correlation between classes and colors and is balanced.

Corrupted CIFAR-10 has been introduced by Hendrycks & Dietterich (2019). There are 50,000 training RGB images and 10 classes. The bias here stems from the fact that each image is corrupted with a specific noise (e.g., Gaussian blur, salt and pepper noise, etc.). Specifically, each class has a privileged type of noise under which it is observed during training (e.g., most of car images are corrupted with motion blur). There are two versions of the dataset, namely Corrupted CIFAR-10[1] and Corrupted CIFAR-10[2]: in the two versions different types of noise affecting data are present.

4.2 Realistic bias: Waterbirds and Bias Action Recognition

We tested our method on real images datasets using Waterbirds and Bias Action Recognition (BAR). Waterbirds has been introduced by Sagawa et al. (2019) and combines bird photos from the Caltech-UCSD Birds-200-2011 (CUB) dataset (Welinder et al. (2010)) with background images from the

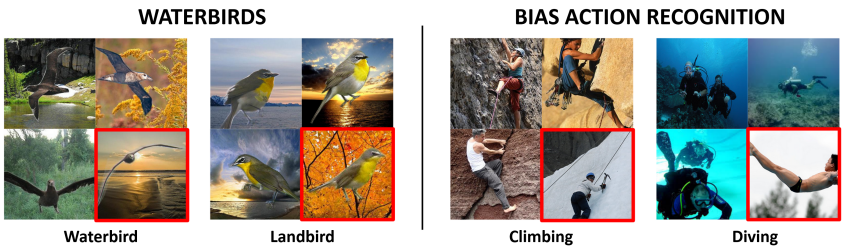


Figure 3: Examples of biased training data and unbiased data (with red boundary) from Waterbirds and BAR.

Places dataset (Zhou et al. (2018)). There are 4,795 training images and the goal is to distinguish two classes, namely *landbird* and *waterbird*. The bias here is represented by the background of the images: most landbirds are observed with land background while most waterbirds are observed with a marine background.

BAR has been introduced by Nam et al. (2020) as a realistic benchmark to test model's debiasing capabilities. It is constructed using several data sources and consists of 1,941 photos of people performing several actions, and the task is to distinguish them in 6 classes: Climbing, Diving, Fishing, Racing, Throwing and Vaulting. The bias arises from the context in which action photos are observed at training: for instance, climbing actions are performed in a dry mountain scenario at training time, whereas in the test set, they are set in a snowy environment. For more details, readers can refer to the original paper.

4.3 Performances

We report the performance of our approach on the different benchmarks above mentioned; accuracy is the metric adopted. Since we deal with biased training data and balanced data in testing, we report both accuracies on the testing subset of unbiased samples only, those under-represented in the training data, as well as over the entire test set (biased + unbiased), to assess how much we lose on the biased samples. In fact, as we learn features having higher generalization capacity, spurious correlations are likely less exploited to classify biased examples, and this may cause a drop in performance on such samples.

For Colored-MNIST, our network f_θ is an MLP with 3 hidden layers with 100 neurons each. We used pre-trained ResNet-18 (on ImageNet (Krizhevsky et al. (2012))) as a backbone for Corrupted CIFAR-10 and BAR, and pre-trained ResNet-50 as backbone for Waterbirds. We remove the last layer from such backbones, adding a 2-layer MLP head on top of it.

The meta-parameter θ^* is computed only for the last two fully connected layers while the backbone is trained with only the contribution of the weighted ERM in Eq. 5 ($\zeta = 0$). We set the learning rate $\eta = 0.001$ for all datasets with batch size$= 256$ on synthetic biased data $= 128$ for realistic bias. We used Adam (Kingma & Ba (2015)) as optimizer. All the experiments comply the same evaluation protocol used in the competing methods for a fair comparison. All implementation details are reported in the Appendix.

Results on the synthetic bias datasets. We first show the results on synthetically biased datasets in Tables 1 and 2, reporting the overall average accuracy and the one for unbiased samples only, respectively. We compare against two baselines, a model trained by Empirical Risk Minimization (ERM) and our method with $\zeta = 0$, which cancels out the contribution of the regularization brought by the outer loop step in Eq. 5. This second baseline only weighs the contributions of the two splits found via pseudo-labeling. We also compare our approach with several methods to learn unbiased representations, either using annotation for the bias or not. For the methods requiring explicit knowledge of the bias, we consider REPAIR (Li & Vasconcelos (2019)), which does sample upweighting, and Group-DRO (Sagawa et al. (2019)), which tackles the problem using robust optimization. We finally report the performance of our direct competitor, Learning from Failure (LfF) (Nam et al. (2020)), which is able to learn a debiased model without exploiting the labeling of the bias.

We consider different ratios of the bias (ranging from 95% up to 99.5%) as in Nam et al. (2020). This ratio indicates the actual percentage of the dataset belonging to $\mathcal{D}_{bias}$ and $\mathcal{D}_{unbias}$. Since we do not know such ratio, in all experiments, we fix the hyper-parameter $\gamma = 0.85$, i.e. we consider 85% of the training data as biased, and therefore assigned to $\hat{\mathcal{D}}_{bias}$, and the remaining 15% to $\hat{\mathcal{D}}_{unbias}$. Since γ is a sensitive parameter, we provide an ablation analysis in which we show how

Dataset	Bias ratio	**ERM**	**Li et al. (2019)**	**Sagawa et al. (2019)**	**Nam et al. (2020)**	**Ours,** $\zeta = 0$	**Ours,** $\zeta = 10$
Colored-MNIST	95%	77.6 ± 0.44	82.5 ± 0.59	84.5 ± 0.46	85.3 ± 0.94	82.3 ± 0.99	**89.3 ± 1.02**
	98%	62.3 ± 1.47	72.9 ± 1.47	76.3 ± 1.53	80.5 ± 0.45	73.8 ± 0.87	**83.4 ± 0.97**
	99%	50.3 ± 0.16	67.3 ± 1.69	71.3 ± 1.76	74.0 ± 2.21	68.3 ± 0.98	**81.6 ± 0.96**
	99.5%	35.3 ± 0.13	56.4 ± 3.74	59.7 ± 2.73	63.4 ± 1.97	57.1 ± 1.05	**72.2 ± 0.87**
Corrupted CIFAR-10^1	95%	45.2 ± 0.22	48.7 ± 0.71	53.1 ± 0.53	59.9 ± 0.16	54.3 ± 1.40	**63.3 ± 1.11**
	98%	30.2 ± 0.77	37.9 ± 0.22	40.2 ± 0.23	49.4 ± 0.78	44.4 ± 0.90	**56.2 ± 0.89**
	99%	22.7 ± 0.97	32.4 ± 0.35	32.1 ± 0.83	41.4 ± 2.34	33.4 ± 0.91	**50.5 ± 0.98**
	99.5%	17.9 ± 0.86	26.3 ± 1.06	29.3 ± 0.11	31.7 ± 1.18	26.1 ± 0.94	**43.3 ± 0.97**
Corrupted CIFAR-10^2	95%	41.3 ± 0.46	54.1 ± 1.01	57.9 ± 0.31	58.6 ± 1.18	53.8 ± 1.21	**62.5 ± 0.91**
	98%	28.3 ± 0.77	44.2 ± 0.84	46.1 ± 1.11	48.7 ± 1.68	43.2 ± 0.96	**55.2 ± 0.98**
	99%	20.7 ± 0.81	38.4 ± 0.26	39.6 ± 1.04	41.3 ± 2.08	37.0 ± 0.99	**49.8 ± 1.01**
	99.5%	17.4 ± 0.85	31.0 ± 0.42	$.342 \pm 0.74$	34.1 ± 2.39	30.6 ± 0.89	**43.6 ± 1.32**

Table 1: **Accuracy on whole test set.** Accuracy (in %) evaluated on biased + unbiased test samples for different bias ratios. Best performance are marked in bold.

Dataset	Bias ratio	**ERM**	**Li et al. (2019)**	**Sagawa et al. (2019)**	**Nam et al. (2020)**	**Ours,** $\zeta = 0$	**Ours,** $\zeta = 10$
Colored-MNIST	95%	75.2 ± 0.87	83.3 ± 1.23	83.1 ± 0.81	85.8 ± 0.66	82.1 ± 0.88	**89.2 ± 1.09**
	98%	58.1 ± 0.56	73.4 ± 0.79	74.3 ± 1.09	80.7 ± 0.56	73.3 ± 0.73	**83.4 ± 0.85**
	99%	44.8 ± 0.84	68.3 ± 0.75	69.6 ± 0.63	74.2 ± 1.94	67.6 ± 0.92	**81.6 ± 0.79**
	99.5%	28.1 ± 0.45	57.3 ± 0.61	57.1 ± 0.78	63.5 ± 1.94	56.8 ± 0.79	**72.1 ± 0.94**
Corrupted CIFAR-10^1	95%	39.4 ± 0.75	50.0 ± 0.89	49.0 ± 0.48	59.6 ± 0.03	54.3 ± 0.89	**63.3 ± 1.10**
	98%	22.6 ± 0.45	38.9 ± 0.64	35.1 ± 0.92	48.7 ± 0.70	44.1 ± 0.83	**56.1 ± 0.92**
	99%	14.2 ± 0.91	33.0 ± 0.57	28.0 ± 0.68	39.5 ± 2.56	32.3 ± 0.84	**49.6 ± 0.85**
	99.5%	10.5 ± 0.28	26.5 ± 0.46	24.4 ± 0.48	28.6 ± 1.25	25.6 ± 0.91	**42.1 ± 0.88**
Corrupted CIFAR-10^2	95%	34.9 ± 0.84	54.5 ± 1.04	54.6 ± 0.61	58.6 ± 1.04	53.6 ± 0.86	**62.3 ± 1.04**
	98%	20.5 ± 0.64	44.6 ± 0.83	42.7 ± 0.77	48.9 ± 1.61	43.8 ± 0.84	**55.5 ± 0.98**
	99%	12.1 ± 0.75	38.8 ± 0.75	37.1 ± 1.22	40.8 ± 2.06	36.4 ± 0.93	**49.7 ± 0.94**
	99.5%	10.0 ± 0.84	31.4 ± 0.53	30.9 ± 0.89	32.0 ± 2.51	29.8 ± 0.91	**43.0 ± 0.85**

Table 2: **Results on unbiased test samples.** Accuracy (in %) evaluated only on the unbiased samples for different bias ratios. Best performance are marked in bold.

the performance changes as γ varies (see Section 4.4 below). Moreover, we set $\zeta = 10$ throughout all the experiments: in Appendix A, we report an ablation about this parameter.

We can observe consistent better results with respect to the competitors, for all datasets and all possible bias ratios. Interestingly, the difference from the baselines increases as the dataset is more biased (higher bias ratio): this indicates that our method is more effective as the bias is more severe. The weighted ERM ($\zeta = 0$) is already a strong baseline that surpasses, in some cases, the former debiasing methods. Please, note that for both the unbiased samples and, in average, over the whole test set, the improvement is significant by a large margin, with a minimum of about 4% up to 17%. This denotes that our approach is not only better at generalizing over unbiased samples, but also maintains high accuracy over the biased examples.

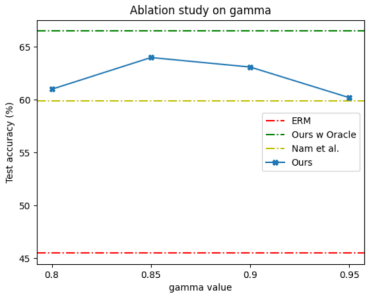
Results on the realistic biased datasets. In these trials, we still compare against the ERM baseline and Group DRO, as supervised method as before, and four unsupervised algorithms, LfF (Nam et al. (2020)), CVaR DRO (Levy et al. (2020)), ReBias (Bahng et al. (2020)), and JTT (Liu et al. (2021)). Performances are reported in Figure 4(a). For these datasets, we remind that we do not have the full control of the bias ratios. Differently from Colored MNIST and Corrupter CIFAR-10, which have a balanced test set, Waterbirds test set is imbalanced: the goal is to increase the accuracy on the unbiased samples without dropping the overall test accuracy, i.e. finding a good trade-off between generalizing to unbiased samples keeping high performance on biased data as well. We score favorably with respect to other unsupervised methods: we reach the second highest accuracy on biased samples, statistically similar to JTT (which scores slightly higher), but we outperform it over the whole test set, in average. We show also competitive performance against the supervised method Group DRO: without using any bias supervision, our method surpasses its average test accuracy even if the accuracy on biased data results lower (owing to the supervision). Concerning the BAR dataset, our method outperforms all other competitors by a considerable margin. Since there is no ground-truth for the bias, we only reported the average accuracy on the whole test set.

4.4 Ablation Study

We conducted an ablation analysis using Corrupted CIFAR-10^1 (bias ratio$= 95\%$), to assess the contribution of each step characterizing our approach. First, we want to test the robustness of the

Method	Bias supervision	Waterbirds		BAR
		Avg. Acc.	Acc. Unbias	Avg. Acc.
ERM	No	97.3%	72.6%	53.5%
CVaR DRO (Levy et al. (2020))	No	96.0%	75.9%	-
LfF (Nam et al. (2020))	No	91.2%	78.0%	62.9%
ReBias (Bahng et al. (2020))	No	-	-	59.7%
JTT (Liu et al. (2021))	No	93.3%	86.7%	-
Group DRO (Sagawa et al. (2019)) (Supervised)	Yes	93.5%	91.4%	-
Ours ($\zeta = 10$)	No	94.1%	86.5%	64.3%

(a)



(b)

Figure 4: **Results on Waterbirds and BAR.** (a) Performance on the whole (Avg.) and unbiased (Unbias) only test set: comparisons with baseline, unsupervised and supervised methods (see text for discussion). **Ablation on γ.** (b) We set $\gamma = 0.8, 0.85, 0.9, 0.95$ and reported the related final accuracy. We compare with ERM baseline, Nam et al. (2020) and our method using the ground-truth bias knowledge as an oracle, i.e., imposing $\hat{\mathcal{D}}_{bias} = \mathcal{D}_{bias}$ and $\hat{\mathcal{D}}_{unbias} = \mathcal{D}_{unbias}$.

Set 1	**Set 2**	**Bias ratio** 95%		98%		99%		99.5%	
		Acc. all	Acc. unbias	Acc. all	Acc. unbias	Acc. all	Acc. unbias	Acc. all	Acc. unbias
No augmentation		58.8%	55.3%	46.1%	41.5%	40.0%	34.8%	33.6%	27.1%
$\hat{\mathcal{D}}_{bias}$	$\hat{\mathcal{D}}_{bias}$	35.2%	29.7%	34.0%	28.4%	32.9%	26.7%	32.0%	27.5%
$\hat{\mathcal{D}}_{unbias}$	$\hat{\mathcal{D}}_{unbias}$	60.2%	63.1%	54.1%	55.3%	48.4%	48.7%	40.4%	42.5%
$\hat{\mathcal{D}}_{bias}$	$\hat{\mathcal{D}}_{unbias}$	**63.8%**	**63.3%**	**56.4%**	**55.9%**	**50.9%**	**49.4%**	**43.1%**	**42.7%**

Table 3: **Ablation analysis on the augmentation strategies.** We report the accuracy resulting from different augmentation strategies and no augmentation, by varying the bias ratio. Our strategy results the winner over all the other mixing policies.

classification performance towards the choice of the hyper-parameter γ that governs the amount of data that we assign to the pseudo-labeled subsets. Results can be seen in Figure 4(b): we observe that by varying γ from 80% to 95%, the final accuracy does not change sensibly, meaning that the initial training of the network f_ϕ is not a critical step as long as the biased training samples can be learnt faster than the unbiased ones.

Second, we tested different strategies to perform data augmentation in the outer loop step. We combined samples from $\hat{\mathcal{D}}_{bias}$ and $\hat{\mathcal{D}}_{unbias}$ (Eq. 4). In Table 3 we report the results when sampling $\hat{x}_1$, $\hat{x}_2$ from different combinations of the subsets. Mixing both samples from $\hat{\mathcal{D}}_{bias}$ overfits the biased data and results in the worst accuracy, while mixing both samples from $\hat{\mathcal{D}}_{unbias}$ increases the generalization on unbiased samples but provides suboptimal results, especially for the biased subset. Samples from $\hat{\mathcal{D}}_{bias}$ mixed with $\hat{\mathcal{D}}_{unbias}$ corresponds to our policy, which provides the best performance. We also report the case in which no augmentation is performed (first row), i.e. x_{mix}, y_{mix} are just drawn from $\mathcal{D}$.

5 Conclusions

We proposed a novel solution for the problem of unsupervised debiasing using a meta-learning strategy. After having subdivided by a pseudo-labeling method the training dataset into two subsets of biased and unbiased samples, we treated them as tasks to be learned a bi-level optimization algorithm. The key idea aimed at better generalization is the mixing of the two subsets to provide the model with unseen data that can break the learning of the spurious correlations between data and class labels. As future directions, we point out two main problems to be addressed. First, designing more robust strategies to perform the pseudo-labeling stage to reach and perhaps even surpass the performance using ground-truth bias knowledge. Second, finding more refined ways of combining biased and unbiased samples to allow the model to reach a better generalization.

REFERENCES

Mohsan Alvi, Andrew Zisserman, and Christoffer Nellåker. Turning a blind eye: Explicit removal of biases and variation from deep neural network embeddings. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pp. 0–0, 2018.

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*, pp. 528–539, 2020.

Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/647bba344396e7c8170902bcf2e15551-Paper.pdf.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135. PMLR, 06–11 Aug 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *CoRR*, abs/1903.12261, 2019. URL http://arxiv.org/abs/1903.12261.

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*, pp. 9012–9020, 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25, pp. 1097–1105. Curran Associates, Inc., 2012.

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

Daniel Levy, Yair Carmon, John C. Duchi, and Aaron Sidford. Large-scale methods for distributionally robust optimization, 2020.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Learning to generalize: Meta-learning for domain generalization. *CoRR*, abs/1710.03463, 2017. URL http://arxiv.org/abs/1710.03463.

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 (CVPR)*, pp. 9572–9581, 2019.

Yiying Li, Yongxin Yang, Wei Zhou, and Timothy Hospedales. Feature-critic networks for heterogeneous domain generalisation. In *The Thirty-sixth International Conference on Machine Learning*, 2019.

Evan Zheran 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. *CoRR*, abs/2107.09044, 2021. URL https://arxiv.org/abs/2107.09044.

Daniel Moyer, Shuyang Gao, Rob Brekelmans, Greg Ver Steeg, and Aram Galstyan. Evading the adversary in invariant representation. *CoRR*, abs/1805.09458, 2018. URL http://arxiv.org/abs/1805.09458.

Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: Training debiased classifier from biased classifier. *Advances on Neural Information Processing systems (NeurIPS)*, 2020.

Ruggero Ragonesi, Riccardo Volpi, Jacopo Cavazza, and Vittorio Murino. Learning unbiased representations via mutual information backpropagation, 2020.

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. *CoRR*, abs/1911.08731, 2019. URL http://arxiv.org/abs/1911.08731.

Haohan Wang, Zexue He, Zachary C Lipton, and Eric P Xing. Learning robust representations by projecting superficial statistics out. *The International Conference on Learning Representations (ICLR)*, 2019.

P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *CoRR*, abs/1710.09412, 2017. URL http://arxiv.org/abs/1710.09412.

Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, 2018. doi: 10.1109/TPAMI.2017.2723009.

A ABLATION STUDY ON ζ

We set different values of ζ, i.e., $[0, 1, 10, 100, 1000]$ on the synthetic biased dataset Corrupted CIFAR-10[1] with bias ratio$= 95\%$, and we show in Figure 5 the accuracy on the whole test set and on the unbiased samples. We note that there is a large range, $1 < \zeta < 100$, in which the accuracy is reaching high values for both subsets (please, note the logarithmic scale in the x axis). This empirically shows that ζ is not so a sensitive parameter with respect to the proposed strategy, and for this reason, we fix $\zeta = 10$ in all our experiments.

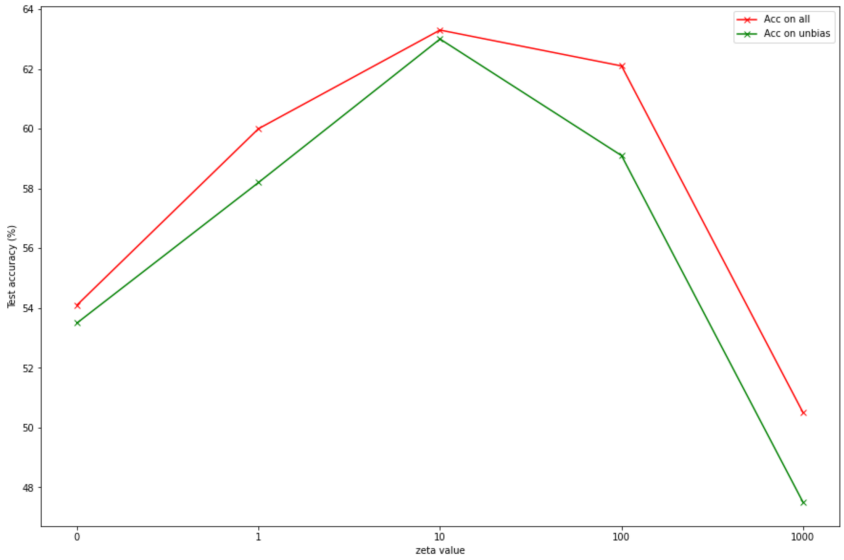


Figure 5: Test accuracy for unbiased samples (Green) and generic samples (Red) for different values of ζ. X axis is in logarithmic scale. For $\zeta = 0.0$ we have the weighted ERM of Eq. 5.

B IMPLEMENTATION DETAILS

We report some additional details regarding the experimental section: we followed prior works (Nam et al. (2020); Liu et al. (2021); Sagawa et al. (2019)) in order to have results as much comparable as possible.

Colored MNIST. We used an MLP with 3 hidden layers with 100 neurons each as f_θ. We set learning rate $\eta = 0.001$, batch size$= 256$, hyperparameters $\gamma = 0.85$, $\zeta = 10$ and trained for $K = 100$ epochs. We used Adam (Kingma & Ba (2015)) as optimizer.

Corrupted CIFAR-10[1], 2**.** We used the Pytorch implementation of ResNet-18 (He et al. (2015)) with pre-training on ImageNet (Krizhevsky et al. (2012)). We removed the last layer and added a 2-layers MLP with 256 neurons in the hidden layer on top of the backbone. We set learning rate $\eta = 0.001$, batch size$= 256$, hyperparameters $\gamma = 0.85$, $\zeta = 10$ and trained for $K = 100$ epochs. We used Adam as optimizer. We used random crops as data augmentation as in Nam et al. (2020). We compute θ^* only for the MLP-head parameters: in other words, the backbone is not involved in the meta-learning process but is trained only with the Weighted ERM contribute of Eq. 5.

Waterbirds. We used pre-trained ResNet-50 (He et al. (2015)) (following Liu et al. (2021)). We removed the last layer and added a 2-layers MLP with 256 neurons in the hidden layer on top of the backbone. We set learning rate $\eta = 0.001$, batch size$= 128$, hyperparameters $\gamma = 0.85$, $\zeta = 10$ and trained for $K = 100$ epochs. We used Adam as optimizer (with weight decay$= 0.0001$) and no data augmentation. We compute θ^* only for the MLP-head parameters as we do for Corrupted CIFAR-10.

BAR. We used pre-trained ResNet-18 (He et al. (2015)) from which we removed the last layer and added a 2-layers MLP with 256 neurons in the hidden layer on top of the backbone. We set learning

rate $\eta = 0.001$, batch size$= 128$, hyperparameters $\gamma = 0.85$, $\zeta = 10$ and trained for $K = 100$ epochs. We used Adam as optimizer (with weight decay$= 0.0001$) and random resized crops as data augmentation as in Nam et al. (2020) We compute θ^* only for the MLP-head parameters as we do for Corrupted CIFAR-10.