Last Layer Re-Training is Sufficient for Robustness to Spurious Correlations

Polina Kirichenko*, **Pavel Izmailov***, **Andrew Gordon Wilson**
New York University

Abstract

Neural network classifiers can largely rely on simple spurious features, such as backgrounds, to make predictions. However, even in these cases, we show that they still often learn core features associated with the desired attributes of the data, contrary to recent findings. Inspired by this insight, we demonstrate that simple last layer retraining can match or outperform state-of-the-art approaches on spurious correlation benchmarks, but with profoundly lower complexity and computational expenses. Moreover, we show that last layer retraining on large ImageNet-trained models can also significantly reduce reliance on background and texture information, improving robustness to covariate shift, after only minutes of training on a single GPU.

1 Introduction

Realistic datasets in computer vision are riddled with *spurious correlations* — patterns that are predictive of the target in the train data, but that are irrelevant to the true labeling function. For example, most of the images labeled as butterfly on ImageNet also show flowers [90], and most of the images labeled as tench show a fisherman holding the tench [12]. Deep neural networks rely on these spurious features, and consequently degrade in performance when tested on examples where the spurious correlations break, for example, on images with unusual background contexts [25, 82, 7]. In an especially alarming example, CNNs trained to recognize pneumonia were shown to rely on hospital-specific metal tokens in the chest X-ray scans, instead of features relevant to pneumonia [103]. Furthermore, spurious correlations can disproportionately affect different population groups (each with their own data distribution), leading to poor performance on the minority groups and unfair decisions [13, 39, 10, 94, 32, 20, 17, 79].

In this paper, we investigate what features are in fact learned on datasets with spurious correlations. We find that even when neural networks appear to heavily rely on spurious features and perform poorly on minority groups where the spurious correlation is broken, they still learn the core features sufficiently well. These core features, associated with the semantic structure of the problem, are learned even in cases when the spurious features are much simpler than the core features (see Section 4.2) and in some cases even when no minority group examples are present in the training data (see Section 6)! While both the relevant and spurious features are learned, the spurious features can be highly weighted in the final classification layer of the model, leading to poor predictions on the minority groups.

Inspired by these observations, we propose *Deep Feature Reweighting* (DFR), a simple and effective method for improving worst-group accuracy of neural network classifiers in the presence of spurious features. We illustrate DFR in Figure 1. In DFR, we simply retrain the last layer of a classification model trained with standard Empirical Risk Minimization

*Equal contribution.

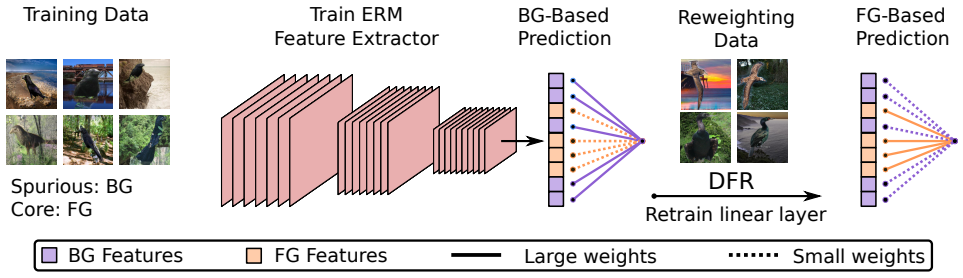


Figure 1: **Deep feature reweighting (DFR).** An illustration of the DFR method on the Waterbirds dataset, where the background (BG) is spuriously correlated with the foreground (FG). Standard ERM classifiers learn both features relevant to the background and the foreground, and weight them in a way that the model performs poorly on images with confusing backgrounds. With DFR, we simply reweight these features by retraining the last linear layer on a small dataset where the backgrounds are not spuriously correlated with the foreground. The resulting DFR model primarily relies on the foreground, and performs much better on images with confusing backgrounds.

(ERM), using a small set of *reweighting* data where the spurious correlation does not hold. DFR achieves state-of-the-art performance on popular spurious correlation benchmarks by simply reweighting the features of a trained ERM classifier, with no need to re-train the feature extractor. Moreover, we show that DFR can be used to reduce reliance on background and texture information and improve robustness to certain types of covariate shift in large-scale models trained on ImageNet, by simply retraining the last layer of these models. We note that the reason DFR can be so successful is because standard neural networks *are* in fact learning core features, even if they do not primarily rely on these features to make predictions, contrary to recent findings [38, 87]. Since DFR only requires retraining a last layer, amounting to logistic regression, it is extremely simple, easy to tune and computationally inexpensive relative to the alternatives, yet can provide state-of-the-art performance. Indeed, DFR can reduce texture bias and improve robustness of large ImageNet trained models, in only minutes on a single GPU.

Our code is available at `github.com/PolinaKirichenko/deep_feature_reweighting`.

2 Problem Setting

We consider classification problems, where we assume that the data consists of several *groups* $\mathcal{G}_i$, which are often defined by a combination of a label and spurious attribute. Each group has its own data distribution $p_i(x, y)$, and the training data distribution is a mixture of the group distributions $p(x, y) = \sum_i \alpha_i p_i(x, y)$, where α_i is the proportion of group $\mathcal{G}_i$ in the data. For example, in the Waterbirds dataset [85], the task is to classify whether an image shows a landbird or a waterbird. The groups correspond to images of waterbirds on water background ($\mathcal{G}_1$), waterbirds on land background ($\mathcal{G}_2$), landbirds on water background ($\mathcal{G}_3$) and landbirds on land background ($\mathcal{G}_4$). See Figure 3 for a visual description of the Waterbirds data. We will consider the scenario when the groups are not equally represented in the data: for example, on Waterbirds the sizes of the groups are 3498, 184, 56 and 1057, respectively. The larger groups $\mathcal{G}_1, \mathcal{G}_4$ are referred to as *majority groups* and the smaller $\mathcal{G}_2, \mathcal{G}_3$ are referred to as minority groups. As a result of this heavy imbalance, the background becomes a *spurious feature*, i.e. it is a feature that is correlated with the target on the train data, but it is not predictive of the target on the minority groups. Throughout the paper we will discuss multiple examples of spurious correlations in both natural and synthetic datasets.

In this paper, we study the effect of spurious correlations on the features learned by standard neural networks, and based on our findings propose a simple way of reducing the reliance on spurious features assuming access to a small set of data where the groups are equally represented.

3 Related Work

Spurious correlations are ubiquitous in real-world applications. There is hence an active research effort towards understanding and reducing their effect on model performance. Here, we review the key methods and related work in this area.

Feature learning in the presence of spurious correlations. The poor performance of neural networks on datasets with spurious correlations inspired research in understanding when and how the spurious features are learned. Geirhos et al. [25] provide a detailed survey of the results in this area. Several works explore the behavior of maximum-margin classifiers, SGD training dynamics and inductive biases of neural network models in the presence of spurious features [66, 76, 78]. Shah et al. [87] show empirically that in certain scenarios neural networks can suffer from *extreme simplicity bias* and rely on simple spurious features, while ignoring the core features; in Section 4.2 we revisit these problems and provide further discussion. Hermann and Lampinen [38] and Jacobsen et al. [45] also show synthetic and natural examples, where neural networks ignore relevant features, and Scimeca et al. [86] explore which types of shortcuts are more likely to be learned. Kolesnikov and Lampert [50] on the other hand show that on realistic datasets core and spurious features can often be distinguished from the latent representations learned by a neural network in the context of object localization.

In an independent and concurrent work, Rosenfeld et al. [83] show that standard ERM learns high quality representations in the context of domain generalization, and by training the last layer on the target domain it is possible to achieve strong results. The observations in our work are complimentary, as they focus on different settings.

Spurious correlations in natural image datasets. Multiple works demonstrated that natural image datasets contain spurious correlations that hurt neural network models [50, 100, 89, 3, 90, 91, 64]. Notably, Geirhos et al. [27] demonstrated that ImageNet-trained CNNs are biased towards texture rather than shape of the objects. Follow-up work explored this texture bias and showed that despite being texture-biased, CNNs still often represent information about the shape in their feature representations [37, 44]. In Section 7 we show that it is possible to reduce the reliance of ImageNet-trained models on background context and texture information by retraining just the last layer of the model.

Improving group robustness. The methods achieving the best worst-group performance typically build on the distributionally robust optimization (DRO) framework, where the worst-case loss is minimized instead of the average loss [8, 40, 85, 70, 106]. Notably, Group-DRO [85], which optimizes a soft version of the worst-group loss holds state-of-the-art results on multiple benchmarks with spurious correlations. Several methods have been proposed for the scenario where group labels are not known, and need to be inferred from the data; these methods typically train a pair of networks, where the first model is used to identify the challenging minority examples and define a weighted loss for the second model [58, 67, 102, 96, 107, 18, 19]. Other works proposed semi-supervised methods for the scenario where the group labels are provided for a small fraction of the train datapoints [92, 68]. Idrissi et al. [43] recently showed that with careful tuning simple approaches such as data subsampling and reweighting can provide competitive worst-group performance. We note that all of the methods described above use a validation set with a high representation of minority groups to tune the hyper-parameters and optimize worst-group performance.

Another group of papers proposes regularization techniques to learn diverse solutions on the train data, focusing on different groups of features [95, 56, 71, 76]. Xu et al. [101] show how to train *orthogonal classifiers*, i.e. classifiers invariant to given spurious features in the data. Other papers proposed methods based on meta-learning the weights for a weighted loss [80] and group-agnostic adaptive regularization [15, 14]. Related methods have been developed in several areas of machine learning, such as ML Fairness [22, 30, 47, 77, 2, 46], domain adaptation [23, 24] and domain generalization [9, 65, 57, 29, 84] including works on Invariant Risk Minimization and causality [75, 4, 53, 5].

Transfer learning. Transfer learning [72, 88] is an extremely popular framework in modern machine learning. Multiple works demonstrate its effectiveness in computer vision [e.g. 28, 41, 34, 93, 61, 49, 60], and study when and why transfer learning can be effective [104, 69, 1, 51, 54]. Bommasani et al. [11] provide a comprehensive discussion of modern transfer learning with large-scale models. While algorithmically our proposed DFR method is related to transfer learning, it has a different motivation and works on different problems. We discuss the relation between DFR and transfer learning in detail in Section 5.

Our work contributes to understanding how features are learned in the presence of spurious correlations. We show that even in the cases when ERM-trained models rely heavily on spurious features and underperform on the minority groups, they often still learn high quality representations of the core features. Based on this insight, we propose DFR — a simple and practical method that achieves state-of-the art on popular benchmark datasets by simply reweighting the features in a pretrained ERM classifier.

4 Understanding Representation Learning with Spurious Correlations

In this section we investigate the solutions learned by standard ERM classifiers on datasets with spurious correlations. We show that while these classifiers underperform on the minority groups, they still learn the core features that can be used to make correct predictions on the minority groups.

4.1 Feature learning on Waterbirds data

We first consider the Waterbirds dataset [85] (see Section 2 and Figure 1) which is generated synthetically by combining images of birds from the CUB dataset [98] and backgrounds from the Places dataset [108] (see Sagawa et al. [85] for a detailed description of the data generation process). For the experiments in this section, we generate several variations of the Waterbirds data following the procedure analogous to Sagawa et al. [85]. The *Original* dataset is analogous to the standard Waterbirds data, but we vary the degree of spurious correlation between the background and the target: 50% (*Balanced* dataset), 95% (as in Sagawa et al. [85]) and 100% (no minority group examples in the train dataset). The *FG-Only* dataset contains images of the birds on uniform grey background instead of the Places background, removing the spurious feature. We show examples of datapoints from each variation of the dataset in Appendix Figure 6.

We train a ResNet-50 model with ERM on each of the variations of the data and report the results in Table 1. Following prior work [e.g., 85, 58, 43], we initialize the model with weights pretrained on ImageNet. For the models trained on the Original data, there is a large difference between the mean and worst group accuracy on the Original test data: the model heavily relies on the background information in its predictions, so the performance on minority groups is poor. The model trained without minority groups is especially affected, only achieving 38.4% worst group accuracy. However, surprisingly, we find that all the models trained on the Original data can make much better predictions on the FG-Only test data: if we remove the spurious feature (background) from the inputs at test time, the models make

Train Data	Spurious Corr. (%)	Test Data (Worst/Mean, %)	
		Original	FG-Only
Balanced	50	91.9/97.3	94.7/98.1
Original	95	73.8/90.7	93.7/96.4
Original	100	38.4/71.5	94.6/95
FG-Only	-	75.2/82.5	95.5/98.2

Table 1: **Feature learning on Waterbirds.** ERM classifiers trained on Waterbirds with Original and FG-Only images. All the models trained on the Original data including the model trained without any minority group examples (*Spurious corr.* 100%) underperform on the worst-group accuracy on the Original data, but perform well on the FG-Only data, almost matching the performance of the FG-Only trained model.

predictions based on the core feature (bird), and achieve worst-group[1] accuracy close to 94%, which is only slightly lower than the accuracy of a model trained directly on the FG-Only data and comparable to the accuracy of a model that was trained on balanced train data.

In Appendix A.1, we provide the full details for the experiment in this section, and additionally consider the reverse Waterbirds problem, where instead of predicting the bird type, the task is to predict the background type. We note that prior work did not consider this reverse Waterbirds problem. The results on the reverse Waterbirds problem are analogous to the Waterbirds results presented in Table 1: in the task of predicting the background, the bird type serves as a spurious feature; the model relies on the bird type in its predictions of the background type, but performs well if we remove the bird from the images at test time. Notably, the results between the standard and reverse Waterbirds problems are fairly symmetric, e.g. in the case when the spurious correlation strength is 100%, the worst group accuracy for both problems is below random guess. On the Waterbirds data, neural networks do not appear to be particularly biased towards either the background or the foreground, and treat them equally.

In Appendix A.3, we try to understand how the features from the foreground and the background interact with each other in a Waterbirds-trained neural network. We find a phenomenon that we call *logit additivity*: the class logits on a Waterbirds test image are well approximated as the sum of the logits for the corresponding background image and the logits for the corresponding foreground image. The logit addivity suggests that the background and foreground are processed mostly independently by the network.

In summary, we conclude that while the models trained on the Original data make use of background information to make predictions, they still learn the features relevant to classifying the birds *almost as well as the models trained on the data without spurious correlations*. Later, we will see that we can retrain just the last layer of the Original-trained models and dramatically improve the worst-group performance on the Original data, by emphasizing the features relevant to the bird instead of the background.

4.2 Simplicity bias: Dominoes data

Shah et al. [87] showed that neural networks can suffer from *extreme simplicity bias*, a tendency to completely rely on the simple features, while ignoring similarly predictive (or even more predictive) complex features. In this section, we explore whether the neural networks can still learn the core features in the extreme simplicity bias scenarios, where the spurious features are simple and highly correlated with the target. Following Shah et al. [87]

[1]On the FG-Only data the groups only differ by the bird type, as we remove the background. The difference between mean and worst-group accuracy is because the target classes are not balanced in the training data.

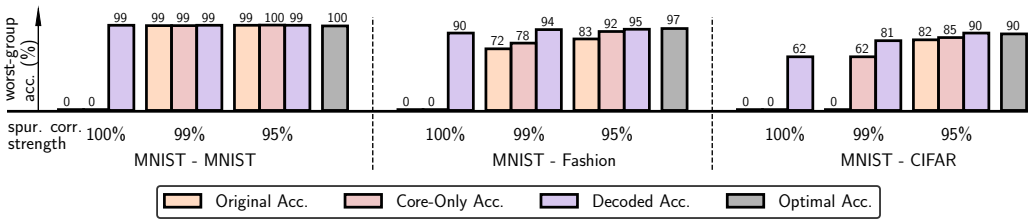


Figure 2: **Feature learning and simplicity bias.** ResNet-20 ERM classifiers trained on Dominoes data with varying levels of spurious correlation between core and spurious features. We show worst-group test accuracy for: Original data, data with only core features present (Core-Only), and accuracy of decoding the core feature from the latent representations of the Original data with logistic regression. We additionally report optimal accuracy: accuracy of a model trained and evaluated on the Core-Only data. Even in cases when the model achieves 0% accuracy on the Original data, the core features can still be decoded from latent representations.

and Pagliardini et al. [71], we consider *Dominoes* binary classification datasets, where the top half of the image shows MNIST digits [55] from classes $\{0, 1\}$, and the bottom half shows MNIST images from classes {7, 9} (*MNIST-MNIST*), Fashion-MNIST [99] images from classes {coat, dress} (*MNIST-Fashion*) or CIFAR-10 [52] images from classes {car, truck} (*MNIST-CIFAR*). In all Dominoes datasets, the top half of the image (MNIST $0-1$ images) presents a linearly separable feature; the bottom half of the image presents a harder to learn feature. See Appendix A.2 for more details about the experimental set-up and datasets, and Appendix Figure 6 for image examples.

We use the simple feature (top half of the images) as a spurious feature and the complex feature (bottom half) as the core feature; we generate datasets with $100\%, 99\%$ and 95% correlations between the spurious feature and the target in train data, while the core feature is perfectly aligned with the target[2]. We then define 4 groups $\mathcal{G}_i$ based on the values of the spurious and core features, where the minority groups correspond to images with top and bottom halfs that do not match. The groups on validation and test are balanced.

We train a ResNet-20 model on each variation of the dataset. In Figure 2 we report the worst group performance for each of the datasets and each spurious correlation strength. In addition to the worst-group accuracy on the Original test data, we report the *Core-Only* worst-group accuracy, where we evaluate the model on datapoints with the spurious top half of the image replaced with a black image. Similarly to Shah et al. [87] and Pagliardini et al. [71], we observe that when the spurious features are perfectly correlated with the target on Dominoes datasets, the model relies just on the simple spurious feature to make predictions and achieves 0% worst-group accuracy. However, with 99% and 95% spurious correlation levels on train, we observe that models learned the core features well, as indicated both by their performance on the Original test data and especially increased performance on Core-Only test data where spurious features are absent.

For reference, on each dataset we also report the *Optimal accuracy*, which is the accuracy of a model trained and evaluated on the Core-Only data. The optimal accuracy provides an upper bound on the accuracy that we can expect on each of the datasets.

Decoding feature representations. The performance on the Original and even Core-Only data might not be fully representative of whether or not the network learned a high-quality representation of the core features. Indeed, even if we remove the MNIST digit from the top half of the image, the network can still primarily rely on the (empty) top half

[2]Shah et al. [87] and Pagliardini et al. [71] only considered Dominoes data where both the simple and complex features are perfectly predictive of the target, corresponding to a 100% spurious correlation.

in its predictions: an empty image may be more likely to come from class 1, which typically has fewer white pixels than class 0. To see how much information about the core feature is contained in the latent representation, we evaluate the *decoded* accuracy: for each problem we train a logistic regression classifier on top of features extracted by the final convolutional layer of the network. We use a balanced[3] validation set to train the logistic regression model, and then report the worst-group accuracy on a test set. In Figure 2, we observe that for MNIST-MNIST and MNIST-FashionMNIST even when the spurious correlation is 100%, reweighting the features leads to high test accuracy. Moreover, on all Dominoes datasets for 99% and 95% spurious correlations level the core features can be decoded with high accuracy and almost match the optimal accuracy. This decoding serves as a basis of our DFR method, which we describe in detail in Section 5.

In Appendix A.2 we report an additional baseline and verify that the model indeed learns a non-trivial representation of the core feature, especially for spurious correlation strengths 99% and 95%.

Relation to prior work. With the same Dominoes datasets that we consider in this section, Shah et al. [87] showed that neural networks tend to rely entirely on the simple features. However, they only considered the 100% spurious correlation strength and accuracy on the Original test data. Our results do not contradict their findings but provide new insights: even in these most challenging cases, the networks still represent information about the complex core feature. Moreover, this information can be decoded to achieve high accuracy on the mixed group examples. Hermann and Lampinen [38] considered a different set of synthetic datasets, showing that in some cases neural networks fail to represent information about some of the predictive features. In particular, they also considered decoding the information about these features from the latent representations and different spurious correlation strengths. Our results add to their observations and show that while it is possible to construct examples where predictive features are suppressed, in many challenging practical scenarios, neural networks learn a high quality representation of the core features relevant to the problem even if they rely on the spurious features. Finally, in a concurrent work, Rosenfeld et al. [83] show that ERM can learn features sufficient for strong performance in the domain generalization setting, adding further support to the results presented in this section.

In summary, we find that, surprisingly, if (1) the strength of the spurious correlation is lower than 100% or (2) the difference in complexity between the core and spurious features is not as stark as on MNIST-CIFAR, *the core feature can be decoded from the learned embeddings with high accuracy.*

5 Deep Feature Reweighting

In Section 4 we have seen that neural networks trained with standard ERM learn multiple features relevant to the predictive task, such as features of both the background and the object represented in the image. Inspired by these observations, we propose *Deep Feature Reweighting (DFR)*, a simple and practical method for improving robustness to spurious correlations and distribution shift.

Let us assume that we have access to a dataset $\mathcal{D} = \{x_i, y_i\}$ which can exhibit spurious correlations. Furthermore, we assume that we have access to a (typically much smaller) dataset $\hat{\mathcal{D}}$, where the groups are represented equally. $\hat{\mathcal{D}}$ can be a subset of the train dataset $\mathcal{D}$, or a separate set of datapoints. We will refer to $\hat{\mathcal{D}}$ as *reweighting dataset*. We start by training a neural network on all of the available data $\mathcal{D}$ with standard ERM without any group or class reweighting. For this stage we do not need any information beyond the training data and labels. Here, we assume that the network consists of a feature extractor (such as a

[3] A balanced validation set contains the same number of datapoints from each of the groups in the dataset.

a sequence of convolutional layers or a vision transformer), followed by a fully-connected classification layer mapping the features to class logits. In the second stage of the procedure, we simply discard the classification head and train a new classification head from scratch on the available balanced data $\hat{\mathcal{D}}$. We use the new classification head to make predictions on the test data. We illustrate DFR in Figure 1.

Notation. We will use notation $\text{DFR}^{\hat{\mathcal{D}}}_{\mathcal{D}}$ to specify the datasets used in each stage of the DFR procedure: $\mathcal{D}$ is used to train the base feature extractor model and $\hat{\mathcal{D}}$ is used to train the last linear layer.

Relation to transfer learning. Algorithmically, DFR is a special case of transfer learning, where $\mathcal{D}$ serves as the source data, and $\hat{\mathcal{D}}$ is the target data [88]. However, the motivation of DFR is different from that of standard transfer learning: in DFR we are trying to correct the behavior of a pretrained model, and reduce the effect of spurious features, while in transfer learning the goal is to learn general features that generalize well to diverse downstream tasks. For example, we can use DFR to reduce the reliance of ImageNet-trained models on the background or texture and improve their robustness to covariate shift (see Section 7), while in standard transfer learning we would typically use a pretrained model as initialization or a feature extractor to learn a good solution on a *new dataset*. In the context of spurious correlations, one would not expect DFR to work as well as it does: the only reason DFR is successful is because, contrary to conventional wisdom, *neural network classifiers are in fact learning substantial information about core features, even when they seem to rely on spurious features to make predictions*. Moreover, in Appendix Table 5 we show that transfer learning with features learned ImageNet does not work nearly as well as DFR on the spurious correlation benchmarks.

6 Feature Reweighting Improves Robustness

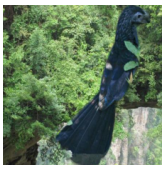
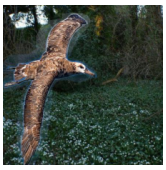
In this section, we evaluate DFR on two benchmark computer vision problems with spurious correlations: Waterbirds and hair-color prediction on CelebA [59].

6.1 Experimental Setup

Data. See Section 2 for a description of the Waterbirds data. On CelebA, the groups are non-blond females ($\mathcal{G}_1$), blond females ($\mathcal{G}_2$), non-blond males ($\mathcal{G}_3$) and blond males ($\mathcal{G}_4$) with sizes $71629, 22880, 66874$, and 1387, respectively. The group $\mathcal{G}_4$ is the minority group, and the gender serves as a spurious feature. See Figure 3 for a visual description of the data.

Baselines. We consider 5 baseline methods that work under different assumptions on the information available at the training time. *Empirical Risk Minimization* (*ERM*) represents conventional training without any procedures for improving worst-group accuracies. *Just Train Twice* (*JTT*) [58] is a method that detects the minority group examples on train data, only using group labels on the validation set to tune hyper-parameters. *Group-DRO* [85] is a state-of-the-art method, which uses group information on train and adaptively upweights worst-group examples during training. *SUBG* is ERM applied to a random subset of the data where the groups are equally represented, which was recently shown to be a surprisingly strong baseline [43]. Finally, *Spread Spurious Attribute* (*SSA*) is a method that attempts to fully exploit the group-labeled validation data with a semi-supervised approach that propagates the group labels to the the training data where group information is not available. We discuss the assumptions on the data for each of these baselines in Appendix B.1.

DFR versions. We consider three variations of $\text{DFR}^{\hat{\mathcal{D}}}_{\mathcal{D}}$, which differ by the way we construct the training dataset $\mathcal{D}$ and the balanced set $\hat{\mathcal{D}}$. In DFR^{Tr}_{Tr} , we use a random group-balanced subset of the train data as $\hat{\mathcal{D}}$, i.e. we keep all of the data from the smallest group, and subsample the data from the other groups to the same size. In DFR^{Val}_{Tr} , we use a balanced

Waterbirds dataset				
	$\mathcal{G}_1$	$\mathcal{G}_2$	$\mathcal{G}_3$	$\mathcal{G}_4$
Image Examples				
Description	landbird on land	landbird on water	waterbird on land	waterbird on water
Class Label	0	0	1	1
# Train data	3498 (73%)	184 (4%)	56 (1%)	1057 (22%)
# Val data	467	466	133	133
Target: bird type;	**Spurious feature**: background type;		**Minority**: $\mathcal{G}_2, \mathcal{G}_3$	

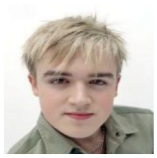
CelebA hair color dataset				
	$\mathcal{G}_1$	$\mathcal{G}_2$	$\mathcal{G}_3$	$\mathcal{G}_4$
Image Examples				
Description	Non-blond woman	Non-blond man	Blond woman	Blond man
Class Label	0	0	1	1
# Train data	71629 (44%)	66874 (41%)	22880 (14%)	1387 (1%)
# Val data	8535	8276	2874	182
Target: hair color;	**Spurious feature**: gender;		**Minority**: $\mathcal{G}_4$	

Figure 3: **Waterbirds and CelebA data.** Dataset descriptions and example images from each group on Waterbirds and CelebA datasaets.

subset of the validation data available for each of the problems. For both $\text{DFR}^{\text{Tr}}_{\text{Tr}}$ and $\text{DFR}^{\text{Val}}_{\text{Tr}}$ we use the standard training dataset (with group imbalance) as $\mathcal{D}$. Finally, with $DFR^{Tr}_{Tr\text{-}NM}$ (NM stands for "No Minority") we use a random group-balanced subset of the train data as $\hat{\mathcal{D}}$, but remove the minority groups ($\mathcal{G}_2, \mathcal{G}_3$ on Waterbirds and $\mathcal{G}_4$ on CelebA) from the data $\mathcal{D}$ used to train the feature extractor. In order to make use of more of the available data, we train logistic regression 10 times using different random balanced subsets of the data, and average the weights of the learned models. Here we can safely average the weights of the learned models because the models are linear. By averaging, we are able to incorporate more of the majority group data without under-weighting the minority groups. Full details are available in Appendix B.

Hyper-parameters. For all DRF variations, the size of the reweighting set $\hat{\mathcal{D}}$ is small relative to the number of features produced by the feature extractor (2048). For this reason, we use ℓ_1-regularizaion to allow the model to learn sparse solutions and drop irrelevant features. For $\text{DFR}^{\text{Val}}_{\text{Tr}}$ and $\text{DFR}^{\text{Tr}}_{\text{Tr-NM}}$ *we only tune a single hyper-parameter* — the strength of the regularization term. For $\text{DFR}^{\text{Tr}}_{\text{Tr}}$ we additionally tune separate weights for the classes, because unlike the other DFR variations, in $\text{DFR}^{\text{Tr}}_{\text{Tr}}$ the feature extractor and the last layer are trained on the same minority data, and logistic regression with equal class weights tends to underperform on the minority groups. We tune all the hyper-parameters on the validation data provided with each of the datasets. We note that the prior work methods including Group-DRO, JTT, SUBG and others extensively tune hyper-parameters on the validation

Method	Group Info		Waterbirds		CelebA	
	Train	Val	Worst(%)	Mean(%)	Worst(%)	Mean(%)
ERM	✗	✓	72.6/85.5$_{\pm 1.0}$	97.3	47.2/79.7$_{\pm 3.7}$	95.6
JTT	✗	✓	86.7/85.6$_{\pm 0.2}$	93.3	81.1/75.6$_{\pm 7.7}$	88.0
Group-DRO	✓	✓	91.4/87.1$_{\pm 3.4}$	93.5	88.9/86.9$_{\pm 1.1}$	92.9
SUBG	✓	✓	89.1$_{\pm 1.1}$	-	85.6$_{\pm 2.3}$	-
SSA	✗	✓✓	89.0$_{\pm 0.55}$	92.2$_{\pm 0.87}$	89.8$_{\pm 1.28}$	92.8$_{\pm 0.11}$
Base Model	✗	✓	74.9$_{\pm 2.4}$	98.1$_{\pm 0.1}$	46.9$_{\pm 2.8}$	95.3$_{\pm 0}$
DFR$^{\text{Tr}}_{\text{Tr}}$	✓	✓	90.2$_{\pm 0.8}$	97.0$_{\pm 0.3}$	80.7$_{\pm 2.4}$	90.6$_{\pm 0.7}$
DFR$^{\text{Val}}_{\text{Tr}}$	✗	✓✓	92.9$_{\pm 0.2}$	94.2$_{\pm 0.4}$	88.3$_{\pm 1.1}$	91.3$_{\pm 0.3}$
Base Model trained without minority groups						
Base Model	✗	✓	31.9$_{\pm 3.6}$	96.0$_{\pm 0.2}$	21.7$_{\pm 3.2}$	95.2$_{\pm 0.1}$
DFR$^{\text{Tr}}_{\text{Tr-NM}}$	✓	✓	89.9$_{\pm 0.6}$	94.3$_{\pm 0.5}$	89.0$_{\pm 1.1}$	91.6$_{\pm 0.4}$

Table 2: **Spurious correlation benchmark results.** Worst-group and mean test accuracy of DFR variations and baselines on Waterbirds and CelebA hair color prediction problems. For the ERM, JTT and Group-DRO baselines, we provide the results reported in two papers: Liu et al. [58] in green and Idrissi et al. [43] in blue. For SUBG, we provide the results from Idrissi et al. [43], which only reported worst-group accuracy, and for SSA, we provide the results from Nam et al. [68]. For mean accuracy, we follow Liu et al. [59] and Sagawa et al. [85] and weight the group accuracies according to their prevalence in the training data. The Group Info column shows whether group labels are available to the methods on train and validation datasets. For DFR$^{\text{Val}}_{\text{Tr}}$, we use the validation data to train the model parameters (last layer) in addition to hyper-parameter tuning, indicated with ✓✓; SSA also uses the validation set to train the model. DFR is competitive with state-of-the-art on both datasets. For DFR we report the mean±std over 5 independent runs of the method.

data. For DFR$^{\text{Val}}_{\text{Tr}}$ we split the validation in half, and use one half to tune the regularization strength parameter; then, we retrain the logistic regression with the optimal regularization on the full validation set. For more details on the hyper-parameter tuning, see Appendix B.

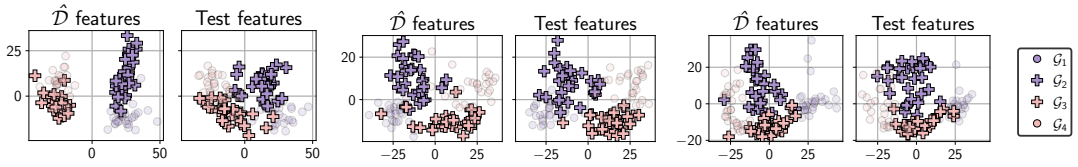
Base model. Following prior work [e.g. 85, 58, 43], we use ResNet-50 initialized with weights pretrained on ImageNet. We select the weight decay and initial learning rate for the base model by choosing the hyper-parameters that produce the base model with the best worst-group accuracy on the validation data. Note that the base model is tuned independently from DFR.

6.2 Results

We report the results for DFR and the baselines in Table 2. All three DFR variations obtain results competitive with state-of-the-art Group-DRO results on the Waterbirds data. On CelebA, DFR$^{\text{Val}}_{\text{Tr}}$ and DFR$^{\text{Tr}}_{\text{Tr-NM}}$ match Group-DRO, while DFR$^{\text{Tr}}_{\text{Tr}}$ performs slightly worse, but still on par with JTT.

In particular, *DFR$^{Tr}_{Tr\text{-}NM}$ matches or outperforms the state-of-the-art Group-DRO by using the same data to train and tune the model*[4]. Notably, DFR$^{\text{Tr}}_{\text{Tr-NM}}$ achieves these results with the features extracted by the network trained *without seeing any examples from the minority groups*! Even without minority groups, ERM models extract the core features sufficiently well to achieve state-of-the-art results.

[4] DFR$^{\text{Tr}}_{\text{Tr-NM}}$ matches the Group-DRO results reported in Sagawa et al. [85] and significantly outperforms the results reported by Idrissi et al. [43].



(a) $\text{DFR}^{\text{Tr}}_{\text{Tr}}$ (b) $\text{DFR}^{\text{Val}}_{\text{Tr}}$ (c) $\text{DFR}^{\text{Tr}}_{\text{Tr-NM}}$

Figure 4: **DFR Variations.** Visualization of the features extracted from the reweighting dataset $\hat{D}$ and the test data for different variations of DFR on the Waterbirds data. We show projections of the 2048-dimensional features on the top-2 principal components extracted from $\hat{D}$. With $\text{DFR}^{\text{Val}}_{\text{Tr}}$ and $\text{DFR}^{\text{Tr}}_{\text{Tr-NM}}$, the distribution of the features for the minority groups $\mathcal{G}_2$ and $\mathcal{G}_3$ does not change between the reweighting and test data, while with $\text{DFR}^{\text{Tr}}_{\text{Tr}}$ we see significant distribution shift.

Moreover, $\text{DFR}^{\text{Val}}_{\text{Tr}}$ achieves similar performance to SSA[5], a method designed to make optimal use of the group information on the validation data. Both methods use the same exact setting and group information to train and tune the model.

To sum up, with DFR we can match the performance of the best available method when group information is available on just the validation set (with $\text{DFR}^{\text{Val}}_{\text{Tr}}$) or on both the train and validation sets (with $\text{DFR}^{\text{Tr}}_{\text{Tr-NM}}$). This state-of-the-art performance is achieved by simply reweighting the features learned by standard ERM, with no need for advanced regularization techniques.

Is ImageNet pretraining necessary? DFR relies on the ability of the feature extractor to learn diverse features, which may suggest that ImageNet pretraining is crucial. In Appendix Table 5, we report the results on the same problems, but training the feature extractor from scratch. We find that on Waterbirds, ImageNet pretraining indeed has a dramatic effect on the performance of DFR as well as the base feature extractor model. However, on CelebA DFR shows strong performance regardless of pretraining. The difference between Waterbirds and CelebA is that Waterbirds contains only $4.8k$ training points, making it difficult to learn a meaningful feature extractor from scratch. Furthermore, on both datasets, finetuning the feature extractor on the target data is crucial: just using the features extracted by an ImageNet-trained model leads to poor results. We note that all the baselines considered in Table 2 use ImageNet pretraining.

6.3 Why is $\text{DFR}^{\text{Tr}}_{\text{Tr-NM}}$ better than $\text{DFR}^{\text{Tr}}_{\text{Tr}}$?

Let us consider the second stage of $\text{DFR}^{\hat{\mathcal{D}}}_{\mathcal{D}}$, where we fix the feature encoder $f(\cdot)$, and train a logistic regression model $\mathcal{L}$ on the dataset $f(\hat{\mathcal{D}})$, where by $f(\hat{\mathcal{D}})$ we denote the dataset with labels from the reweighting dataset $\hat{\mathcal{D}}$ and features from $\hat{\mathcal{D}}$ extracted by f. We then evaluate the logistic regression model $\mathcal{L}$ on the features extracted from the test data, $f(\mathcal{D}_{\text{Test}})$.

Let us use $\hat{\mathcal{M}}$ to denote a minority group in the reweighting dataset $\hat{\mathcal{D}}$, and $\mathcal{M}^{\text{Test}}$ to denote the same minority group in the test data. For $\text{DFR}^{\text{Tr}}_{\text{Tr-NM}}$ and $\text{DFR}^{\text{Val}}_{\text{Tr}}$, the model f is trained without observing any data from $\hat{\mathcal{M}}$ or $\mathcal{M}^{\text{Test}}$. Assuming the datapoints in $\hat{\mathcal{M}}$ and $\mathcal{M}^{\text{Test}}$ are iid samples from the same distirbution, the distribution of features in $f(\hat{\mathcal{M}})$ and $f(\mathcal{M}^{\text{Test}})$ will also be identical.

On the other hand, with $\text{DFR}^{\text{Tr}}_{\text{Tr}}$, the minority group datapoints $\hat{\mathcal{M}}$ are used to train the feature extractor f. In this case, we can no longer assume that the distribution of features $f(\hat{\mathcal{M}})$ will be the same as the distribution of $f(\mathcal{M}^{\text{Test}})$. Consequently, in $\text{DFR}^{\text{Tr}}_{\text{Tr}}$, the logistic regression model $\mathcal{L}$ will be evaluated under *distribution shift*, which makes the problem

[5] $\text{DFR}^{\text{Val}}_{\text{Tr}}$ is slightly better on Waterbirds while SSA is slightly better on CelebA.

much more challenging and leads to inferior performance of $\text{DFR}^{\text{Tr}}_{\text{Tr}}$ on the Waterbirds data. We verify this intuition in Figure 4, where we visualize the feature embeddings for the reweighting dataset $\hat{D}$ and the test data. We see that as we predicted, the distribution of the minority group features coincides between $\hat{\mathcal{D}}$ and test data for $\text{DFR}^{\text{Tr}}_{\text{Tr-NM}}$ and $\text{DFR}^{\text{Val}}_{\text{Tr}}$, while $\text{DFR}^{\text{Tr}}_{\text{Tr}}$ shows significant distribution shift.

What $\hat{\mathcal{D}}$ should be used in practice? In Table 2, the best performance is achieved by $\text{DFR}^{\text{Tr}}_{\text{Tr-NM}}$ and $\text{DFR}^{\text{Val}}_{\text{Tr}}$, which retrain the last layer on data that was not used in training the feature extractor. In practice, we recommend collecting a group-balanced validation set, which can be used both to tune the hyper-parameters and re-train the last layer of the model, as we do in $\text{DFR}^{\text{Val}}_{\text{Tr}}$.

7 Natural Spurious Correlations on ImageNet

In the previous section we focused on benchmark problems constructed to highlight the effect of spurious features. However, computer vision classifiers are known to learn undesirable patterns and rely on spurious features in real-world problems [see e.g. 82, 90, 25]. In this section we explore two prominent shortcomings of ImageNet classifiers: background reliance [100] and texture bias [27].

7.1 Background reliance

Prior work has demonstrated that computer vision models such as ImageNet classifiers can rely on background to make their predictions [105, 81, 89, 100, 90]. Here, we show that it is possible to reduce the background reliance of ImageNet-trained models by simply retraining their last layer with DFR.

Xiao et al. [100] proposed several datasets in the *Backgrounds Challenge* to study the effect of the backgrounds on predictions of ImageNet models. The datasets in the Backgrounds Challenge are based on the ImageNet-9 dataset, a subset of ImageNet structured into 9 coarse-grain classes (see Xiao et al. [100] for details). ImageNet-9 contains $45k$ training images and 4050 validation images. We consider three datasets from the Backgrounds Challenge: (1) *Original* contains the original images; (2) *Mixed-Rand* contains images with random combinations of backgrounds and foregrounds (objects); (3) *FG-Only* contains images showing just the object with a black background. We additionally consider *Paintings-BG* using paintings from Kaggle's `painter-by-numbers` dataset[6] as background for the ImageNet-9 validation data. Finally, we consider the ImageNet-R dataset [35] restricted to the ImageNet-9 classes. See Appendix C for details and Appendix Figure 9 for example images.

We use an ImageNet-trained ResNet-50 as a feature extractor and train DFR with different reweighting datasets. As a *Baseline*, we train DFR on the Original $45k$ training datapoints[7]. We train DFR^{MR} and DFR^{OG+MR} on Mixed-Rand training data and a combination of Mixed-Rand and Original training data respectively. In Figure 5, we report the predictive accuracy of these methods on different validation datasets as a function of the number of Mixed-Rand data observed. We select the observed subset of the data randomly; for $\text{DFR}^{\text{OG+MR}}$, in each case we use the same amount of Mixed-Rand and Original training datapoints. We repeat this experiment with a VIT-B-16 model [21] pretrained on ImageNet-$21k$ and report the results in the Appendix Figure 8.

First, we notice that the baseline model provides significantly better performance on FG-Only (92%) than on the Mixed-Rand (86%) validation set, suggesting that the feature extractor

[6] https://www.kaggle.com/c/painter-by-numbers/

[7] We cannot use the original ImageNet-trained ResNet-50 last layer, as ImageNet-9 has a different set of classes.

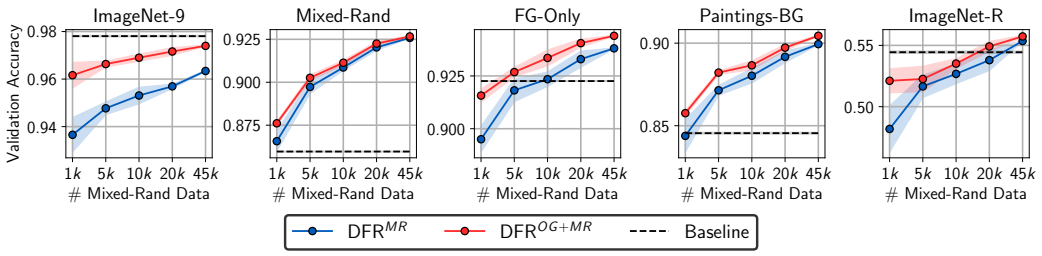


Figure 5: **ImageNet background reliance.** Performance of DFR trained on MixedRand data and MixedRand + Original data on different ImageNet-9 validation splits. All methods use an ImageNet-trained ResNet-50 feature extractor. DFR can reduce background reliance with a minimal drop in performance on the Original data.

learned the features needed to classify the foreground, as well as the background features. With access to Mixed-Rand data, we can reweight the foreground and background features with DFR and significantly improve the performance on mixed-rand, FG-Only and Paintings-BG datasets. At the same time, $\text{DFR}^{\text{OG+MR}}$ is able to mostly maintain the performance on the Original ImageNet-9 data; the small drop in performance is because the background is relevant to predicting the class on this validation set. Finally, on ImageNet-R, DFR provides a small improvement when we use all of $45k$ datapoints; the covariate shift in ImageNet-R is not primarily background-based, so reducing background reliance does not provide a big improvement.

7.2 Texture-vs-shape bias

Geirhos et al. [27] showed that when presented with images with conflicting texture and shape, ImageNet-trained CNNs tend to make predictions based on the texture, while humans usually predict based on the shape of the object. The authors designed the *GST* dataset with conflicting cues, and proposed the term *texture bias* to refer to the fraction of datapoints on which a model (or a human) makes predictions based on texture; conversely, *shape bias* is the fraction of the datapoints on which prediction is made based on the shape of the object. Geirhos et al. [27] showed that it is possible to increase the shape bias of CNNs by training on Stylized ImageNet (*SIN*), a dataset obtained from ImageNet by removing the texture information via style transfer (see Appendix Figure 9 for example images). Using SIN in combination with ImageNet (*SIN+IN*), they also obtained improved robustness to corruptions. Finally, they proposed the *Shape-RN-50* model, a ResNet-50 (RN-50) trained on SIN+IN and finetuned on IN, which outperforms the ImageNet-trained ResNet-50 on in-distribution data and out-of-distribution robustness.

In this section, we explore whether it is possible to change the shape bias of ImageNet-trained models by simply retraining the last layer with DFR. Intuitively, we expect that the standard ImageNet-trained model already learns *both* the shape and texture features. Indeed, Hermann et al. [37] showed that shape information is decodable from the features of ImageNet-trained models on the GST dataset to a certain extent. Here, instead of directly targeting the GST dataset, we apply DFR to the large-scale SIN dataset, and explore both the shape bias and the predictive performance of the resulting models. See Appendix D for details.

In Table 3, we report the shape bias, as well as predictive accuracy of ResNet-50 models trained on ImageNet (*IN*), SIN, IN+SIN and the Shape-RN-50 model, and the DFR models trained on IN, SIN and IN+SIN. The DFR models use an IN-trained ResNet-50 model as a feature extractor.

First, we observe that while the SIN-trained RN-50 achieves a shape-bias of 81.4%, as reported by Geirhos et al. [27], the models trained on combinations of IN and SIN are

Method	Training Data	Shape bias (%)	Top-1 Acc (%) / Top-5 Acc (%)		
			ImageNet	ImageNet-R	ImageNet-C
RN-50	IN	21.4	76.0/92.9	23.8/36.6	39.8/60.7
	SIN	**81.4**	60.3/82.7	26.9/42.1	38.1/59.7
	IN+SIN	34.7	74.6/92.1	**27.6/42.5**	**45.7/67.4**
Shape-RN-50	IN+SIN	20.5	**76.8/93.3**	25.6/39.8	42.3/63.3
DFR	IN	20.8	74.5/92.2	22.6/36.1	36.8/57.2
	SIN	34.0	65.1/85.7	24.6/39.4	36.7/56.7
	IN+SIN	30.6	74.5/91.8	**27.2/42.4**	40.7/61.3

Table 3: **Shape bias.** Shape bias and accuracy on ImageNet validation set variations for ResNet-50 trained on different datasets and DFR with an ImageNet-trained ResNet-50 as a feature extractor. For each metric, we show the best result in bold and underline the best result among DFR methods. By retraining just the last layer with DFR, we can significantly increase the shape bias compared to the base model (21.4% $\rightarrow$ 34% for DFR(SIN)) and improve robustness to covariate shift on ImageNet-R/C.

still biased towards texture. Curiously, the Shape-RN-50 model proposed by Geirhos et al. [27] has almost identical shape bias to a standard IN-trained RN-50! At the same time, Shape-RN-50 outperforms IN-trained RN-50 on all the datasets that we consider, and Geirhos et al. [27] showed that Shape-RN-50 significantly outperforms IN-trained RN-50 in transfer learning to a segmentation problem, suggesting that it learned better shape-based features. The fact that the shape bias of this model is lower than that of an IN-trained RN-50, suggests that the shape bias is largely affected by the last linear layer of the model: even if the model extracted high-quality features capturing the shape information, the last layer can still assign a higher weight to the texture information.

Next, DFR can significantly increase the shape bias of an IN-trained model. $\text{DFR}_{\text{IN}}^{\text{SIN}}$ achieves a comparable shape-bias to that of a model trained from scratch on a combination of IN and SIN datasets. Finally, $\text{DFR}_{\text{IN}}^{\text{IN+SIN}}$ improves the performance on both ImageNet-R and ImageNet-C [36] datasets compared to the base RN-50 model. In the Appendix Table 7 we show similar results for a VIT-B-16 model pretrained on ImageNet-21k and finetuned on ImageNet; there, $\text{DFR}_{\text{IN}}^{\text{IN+SIN}}$ can also improve the shape-bias and performance on ImageNet-C, but does not help on ImageNet-R. To sum up, by reweighting the features learned by an ImageNet-trained model, we can significantly increase its shape bias and improve robustness to certain corruptions. However, to achieve the highest possible shape bias, it is still preferable to re-train the model from scratch, as RN-50(SIN) achieves a much higher shape bias compared to all other methods.

8 Discussion

We have shown that neural networks simultaneously learn multiple different features that can achieve low training loss, including relevant semantic structure, even in the presence spurious correlations. By retraining the last layer of the network with DFR, we can significantly reduce the impact of spurious features and improve worst-group-performance of the models. In particular, DFR achieves state-of-the-art performance on spurious correlation benchmarks, and can reduce the reliance of ImageNet trained models on background and texture information.

Spurious correlations and representation learning. Prior work has often associated poor robustness to spurious correlations with the quality of *representations* learned by the model [4, 6, 84] and suggested that the entire model needs to be carefully trained to

avoid relying on spurious features [e.g. 85, 43, 58, 107, 92, 76, 56]. Our work presents a different view: *representations learned with standard ERM even without seeing any minority group examples are sufficient to achieve state-of-the-art performance on popular spurious correlation benchmarks.* The issue of spurious correlations is not in the features extracted by the models (though the representations learned by ERM can be improved [e.g., 38, 76]), but in the weights assigned to these features. Thus we can simply re-weight these features for substantially improved robustness.

Practical advantages of DFR. DFR is extremely simple, cheap and effective. In particular, DFR has only one tunable hyper-parameter — the strength of regularization of the logistic regression. Furthermore, DFR is highly robust to the choice of base model, as we demonstrate in Appendix Table 6, and does not require early stopping or other highly problem-specific tuning such as in Idrissi et al. [43] and other prior works. Moreover, as DFR only requires re-training the last linear layer of the model, it is also extremely fast and easy to run. For example, we can train DFR on the $1.2M$-datapoint Stylized ImageNet dataset in Section 7 on a single GPU in a matter of minutes, after extracting the embeddings on all of the datapoints, which only needs to be done once. On the other hand, existing methods such as Group-DRO [85] require training the model from scratch multiple times to select the best hyper-parameters, which may be impractical for large-scale problems.

On the need for reweighting data in DFR. Virtually all methods in the spurious correlation literature, even the ones that do not explicitly use group information to train the model, use group-labeled data to tune the hyper-parameters (we discuss the assumptions of different methods in prior work in Appendix B.1). If a practitioner has access to group-labeled data, we believe that they should leverage this data to find a better model instead of just tuning the hyper-parameters. Finally, we note that DFR can be easily combined with methods that automatically estimate group labels [e.g. 58, 18, 92]: we can retrain the last layer using these estimated labels instead of ground truth.

Future work. There are many exciting directions for future research. DFR largely reduces the issue of spurious correlations to a linear problem: how do we train an optimal linear classifier on given features to avoid spurious correlations? In particular, we can try to avoid the need for a balanced reweighting dataset by carefully studying this linear problem and only using the features that are robustly predictive across all of the training data. We can also consider other types of supervision, such as saliency maps [90] or segmentation masks to tell the last layer of the model what to pay attention to in the data. Finally, we can leverage better representation learning methods [76], including self-supervised learning methods [e.g. 16, 33], to further improve the performance of DFR.

Acknowledgements

We would like to thank Micah Goldblum, Wanqian Yang, Marc Finzi, Wesley Maddox, Robin Schirrmeister, Yogesh Balaji, Timur Garipov and Vadim Bereznyuk for helpful comments. This research is supported by an Amazon Research Award, Facebook Research, Capital One, NSF CAREER IIS-2145492, NSF I-DISRE 193471, NIH R01DA048764-01A1, NSF IIS-1910266, and NSF 1922658 NRT-HDR: FUTURE Foundations, Translation, and Responsibility for Data Science. This work was also supported in part through the NYU IT High Performance Computing resources.

References

[1] Abnar, S., Dehghani, M., Neyshabur, B., Sedghi, H.: Exploring the limits of large scale pre-training. arXiv preprint arXiv:2110.02095 (2021)

[2] Agarwal, A., Beygelzimer, A., Dudík, M., Langford, J., Wallach, H.: A reductions approach to fair classification. In: International Conference on Machine Learning, pp. 60–69, PMLR (2018)

[3] Alcorn, M.A., Li, Q., Gong, Z., Wang, C., Mai, L., Ku, W.S., Nguyen, A.: Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4845–4854 (2019)

[4] Arjovsky, M., Bottou, L., Gulrajani, I., Lopez-Paz, D.: Invariant risk minimization. arXiv preprint arXiv:1907.02893 (2019)

[5] Aubin, B., Słowik, A., Arjovsky, M., Bottou, L., Lopez-Paz, D.: Linear unit-tests for invariance discovery. arXiv preprint arXiv:2102.10867 (2021)

[6] Bahng, H., Chun, S., Yun, S., Choo, J., Oh, S.J.: Learning de-biased representations with biased representations. In: International Conference on Machine Learning, pp. 528–539, PMLR (2020)

[7] Beery, S., Van Horn, G., Perona, P.: Recognition in terra incognita. In: Proceedings of the European conference on computer vision (ECCV), pp. 456–473 (2018)

[8] Ben-Tal, A., Den Hertog, D., De Waegenaere, A., Melenberg, B., Rennen, G.: Robust solutions of optimization problems affected by uncertain probabilities. Management Science **59**(2), 341–357 (2013)

[9] Blanchard, G., Lee, G., Scott, C.: Generalizing from several related classification tasks to a new unlabeled sample. Advances in neural information processing systems **24** (2011)

[10] Blodgett, S.L., Green, L., O'Connor, B.: Demographic dialectal variation in social media: A case study of african-american english. arXiv preprint arXiv:1608.08868 (2016)

[11] Bommasani, R., Hudson, D.A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M.S., Bohg, J., Bosselut, A., Brunskill, E., et al.: On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258 (2021)

[12] Brendel, W., Bethge, M.: Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. arXiv preprint arXiv:1904.00760 (2019)

[13] Buolamwini, J., Gebru, T.: Gender shades: Intersectional accuracy disparities in commercial gender classification. In: Conference on fairness, accountability and transparency, pp. 77–91, PMLR (2018)

[14] Cao, K., Chen, Y., Lu, J., Arechiga, N., Gaidon, A., Ma, T.: Heteroskedastic and imbalanced deep learning with adaptive regularization. arXiv preprint arXiv:2006.15766 (2020)

[15] Cao, K., Wei, C., Gaidon, A., Arechiga, N., Ma, T.: Learning imbalanced datasets with label-distribution-aware margin loss. Advances in neural information processing systems **32** (2019)

[16] Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning, pp. 1597–1607, PMLR (2020)

[17] Chouldechova, A.: Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. Big data **5**(2), 153–163 (2017)

[18] Creager, E., Jacobsen, J.H., Zemel, R.: Environment inference for invariant learning. In: International Conference on Machine Learning, pp. 2189–2200, PMLR (2021)

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

[20] Datta, A., Tschantz, M.C., Datta, A.: Automated experiments on ad privacy settings. Proceedings on privacy enhancing technologies **2015**(1), 92–112 (2015)

[21] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)

[22] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., Zemel, R.: Fairness through awareness. In: Proceedings of the 3rd innovations in theoretical computer science conference, pp. 214–226 (2012)

[23] Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: International conference on machine learning, pp. 1180–1189, PMLR (2015)

[24] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.: Domain-adversarial training of neural networks. The journal of machine learning research **17**(1), 2096–2030 (2016)

[25] Geirhos, R., Jacobsen, J.H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., Wichmann, F.A.: Shortcut learning in deep neural networks. Nature Machine Intelligence **2**(11), 665–673 (2020)

[26] Geirhos, R., Narayanappa, K., Mitzkus, B., Thieringer, T., Bethge, M., Wichmann, F.A., Brendel, W.: Partial success in closing the gap between human and machine vision. In: Advances in Neural Information Processing Systems 34 (2021)

[27] Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F.A., Brendel, W.: Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. arXiv preprint arXiv:1811.12231 (2018)

[28] Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp. 1440–1448 (2015)

[29] Gulrajani, I., Lopez-Paz, D.: In search of lost domain generalization. arXiv preprint arXiv:2007.01434 (2020)

[30] Hardt, M., Price, E., Srebro, N.: Equality of opportunity in supervised learning. Advances in neural information processing systems **29** (2016)

[31] Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M.H., Brett, M., Haldane, A., del Río, J.F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., Oliphant, T.E.: Array programming with NumPy. Nature **585**(7825), 357–362 (Sep 2020), doi:10.1038/s41586-020-2649-2, URL https://doi.org/10.1038/s41586-020-2649-2

[32] Hashimoto, T., Srivastava, M., Namkoong, H., Liang, P.: Fairness without demographics in repeated loss minimization. In: International Conference on Machine Learning, pp. 1929–1938, PMLR (2018)

[33] He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. arXiv preprint arXiv:2111.06377 (2021)

[34] He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp. 2961–2969 (2017)

[35] Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., Gilmer, J.: The many faces of robustness: A critical analysis of out-of-distribution generalization. ICCV (2021)

[36] Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. arXiv preprint arXiv:1903.12261 (2019)

[37] Hermann, K., Chen, T., Kornblith, S.: The origins and prevalence of texture bias in convolutional neural networks. Advances in Neural Information Processing Systems **33**, 19000–19015 (2020)

[38] Hermann, K., Lampinen, A.: What shapes feature representations? exploring datasets, architectures, and training. Advances in Neural Information Processing Systems **33**, 9995–10006 (2020)

[39] Hovy, D., Søgaard, A.: Tagging performance correlates with author age. In: Proceedings of the 53rd annual meeting of the Association for Computational Linguistics and the 7th international joint conference on natural language processing (volume 2: Short papers), pp. 483–488 (2015)

[40] Hu, W., Niu, G., Sato, I., Sugiyama, M.: Does distributionally robust supervised learning give robust classifiers? In: International Conference on Machine Learning, pp. 2029–2037, PMLR (2018)

[41] Huh, M., Agrawal, P., Efros, A.A.: What makes imagenet good for transfer learning? arXiv preprint arXiv:1608.08614 (2016)

[42] Hunter, J.D.: Matplotlib: A 2d graphics environment. Computing in Science & Engineering **9**(3), 90–95 (2007), doi:10.1109/MCSE.2007.55

[43] Idrissi, B.Y., Arjovsky, M., Pezeshki, M., Lopez-Paz, D.: Simple data balancing achieves competitive worst-group-accuracy. arXiv preprint arXiv:2110.14503 (2021)

[44] Islam, M.A., Kowal, M., Esser, P., Jia, S., Ommer, B., Derpanis, K.G., Bruce, N.: Shape or texture: Understanding discriminative features in cnns. arXiv preprint arXiv:2101.11604 (2021)

[45] Jacobsen, J.H., Behrmann, J., Zemel, R., Bethge, M.: Excessive invariance causes adversarial vulnerability. arXiv preprint arXiv:1811.00401 (2018)

[46] Khani, F., Raghunathan, A., Liang, P.: Maximum weighted loss discrepancy. arXiv preprint arXiv:1906.03518 (2019)

[47] Kleinberg, J., Mullainathan, S., Raghavan, M.: Inherent trade-offs in the fair determination of risk scores. arXiv preprint arXiv:1609.05807 (2016)

[48] Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S., Willing, C.: Jupyter notebooks – a publishing format for reproducible computational workflows. In: Loizides, F., Schmidt, B. (eds.) Positioning and Power in Academic Publishing: Players, Agents and Agendas, pp. 87 – 90, IOS Press (2016)

[49] Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., Houlsby, N.: Big transfer (bit): General visual representation learning. In: European conference on computer vision, pp. 491–507, Springer (2020)

[50] Kolesnikov, A., Lampert, C.H.: Improving weakly-supervised object localization by micro-annotation. arXiv preprint arXiv:1605.05538 (2016)

[51] Kornblith, S., Shlens, J., Le, Q.V.: Do better imagenet models transfer better? In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 2661–2671 (2019)

[52] Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)

[53] Krueger, D., Caballero, E., Jacobsen, J.H., Zhang, A., Binas, J., Zhang, D., Le Priol, R., Courville, A.: Out-of-distribution generalization via risk extrapolation (rex). In: International Conference on Machine Learning, pp. 5815–5826, PMLR (2021)

[54] Kumar, A., Raghunathan, A., Jones, R., Ma, T., Liang, P.: Fine-tuning can distort pretrained features and underperform out-of-distribution. arXiv preprint arXiv:2202.10054 (2022)

[55] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)

[56] Lee, Y., Yao, H., Finn, C.: Diversify and disambiguate: Learning from underspecified data. arXiv preprint arXiv:2202.03418 (2022)

[57] Li, Y., Tian, X., Gong, M., Liu, Y., Liu, T., Zhang, K., Tao, D.: Deep domain generalization via conditional invariant adversarial networks. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 624–639 (2018)

[58] Liu, E.Z., Haghgoo, B., Chen, A.S., Raghunathan, A., Koh, P.W., Sagawa, S., Liang, P., Finn, C.: Just train twice: Improving group robustness without training group information. In: International Conference on Machine Learning, pp. 6781–6792, PMLR (2021)

[59] Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: Proceedings of the IEEE international conference on computer vision, pp. 3730–3738 (2015)

[60] Maddox, W., Tang, S., Moreno, P., Wilson, A.G., Damianou, A.: Fast adaptation with linearized neural networks. In: International Conference on Artificial Intelligence and Statistics, pp. 2737–2745, PMLR (2021)

[61] Mahajan, D., Girshick, R., Ramanathan, V., He, K., Paluri, M., Li, Y., Bharambe, A., Van Der Maaten, L.: Exploring the limits of weakly supervised pretraining. In: Proceedings of the European conference on computer vision (ECCV), pp. 181–196 (2018)

[62] Marcel, S., Rodriguez, Y.: Torchvision the machine-vision package of torch. In: Proceedings of the 18th ACM international conference on Multimedia, pp. 1485–1488 (2010)

[63] McKinney, W.: Data structures for statistical computing in python. In: van der Walt, S., Millman, J. (eds.) Proceedings of the 9th Python in Science Conference, pp. 51 – 56 (2010)

[64] Moayeri, M., Pope, P., Balaji, Y., Feizi, S.: A comprehensive study of image classification model sensitivity to foregrounds, backgrounds, and visual attributes. arXiv preprint arXiv:2201.10766 (2022)

[65] Muandet, K., Balduzzi, D., Schölkopf, B.: Domain generalization via invariant feature representation. In: International Conference on Machine Learning, pp. 10–18, PMLR (2013)

[66] Nagarajan, V., Andreassen, A., Neyshabur, B.: Understanding the failure modes of out-of-distribution generalization. arXiv preprint arXiv:2010.15775 (2020)

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

[68] Nam, J., Kim, J., Lee, J., Shin, J.: Spread spurious attribute: Improving worst-group accuracy with spurious attribute estimation. In: International Conference on Learning Representations (2022)

[69] Neyshabur, B., Sedghi, H., Zhang, C.: What is being transferred in transfer learning? Advances in neural information processing systems **33**, 512–523 (2020)

[70] Oren, Y., Sagawa, S., Hashimoto, T.B., Liang, P.: Distributionally robust language modeling. arXiv preprint arXiv:1909.02060 (2019)

[71] Pagliardini, M., Jaggi, M., Fleuret, F., Karimireddy, S.P.: Agree to disagree: Diversity through disagreement for better transferability. arXiv preprint arXiv:2202.04414 (2022)

[72] Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on knowledge and data engineering **22**(10), 1345–1359 (2009)

[73] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017)

[74] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)

[75] Peters, J., Bühlmann, P., Meinshausen, N.: Causal inference by using invariant prediction: identification and confidence intervals. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **78**(5), 947–1012 (2016)

[76] Pezeshki, M., Kaba, O., Bengio, Y., Courville, A.C., Precup, D., Lajoie, G.: Gradient starvation: A learning proclivity in neural networks. Advances in Neural Information Processing Systems **34** (2021)

[77] Pleiss, G., Raghavan, M., Wu, F., Kleinberg, J., Weinberger, K.Q.: On fairness and calibration. Advances in neural information processing systems **30** (2017)

[78] Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., Courville, A.: On the spectral bias of neural networks. In: International Conference on Machine Learning, pp. 5301–5310, PMLR (2019)

[79] Rahmattalabi, A., Vayanos, P., Fulginiti, A., Rice, E., Wilder, B., Yadav, A., Tambe, M.: Exploring algorithmic fairness in robust graph covering problems. Advances in Neural Information Processing Systems **32** (2019)

[80] Ren, M., Zeng, W., Yang, B., Urtasun, R.: Learning to reweight examples for robust deep learning. In: International conference on machine learning, pp. 4334–4343, PMLR (2018)

[81] Ribeiro, M.T., Singh, S., Guestrin, C.: " why should i trust you?" explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1135–1144 (2016)

[82] Rosenfeld, A., Zemel, R., Tsotsos, J.K.: The elephant in the room. arXiv preprint arXiv:1808.03305 (2018)

[83] Rosenfeld, E., Ravikumar, P., Risteski, A.: Domain-adjusted regression or: Erm may already learn features sufficient for out-of-distribution generalization. arXiv preprint arXiv:2202.06856 (2022)

[84] Ruan, Y., Dubois, Y., Maddison, C.J.: Optimal representations for covariate shift. arXiv preprint arXiv:2201.00057 (2021)

[85] Sagawa, S., Koh, P.W., Hashimoto, T.B., Liang, P.: Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. arXiv preprint arXiv:1911.08731 (2019)

[86] Scimeca, L., Oh, S.J., Chun, S., Poli, M., Yun, S.: Which shortcut cues will dnns choose? a study from the parameter-space perspective. arXiv preprint arXiv:2110.03095 (2021)

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

[88] Sharif Razavian, A., Azizpour, H., Sullivan, J., Carlsson, S.: Cnn features off-the-shelf: an astounding baseline for recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp. 806–813 (2014)

[89] Shetty, R., Schiele, B., Fritz, M.: Not using the car to see the sidewalk–quantifying and controlling the effects of context in classification and segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8218–8226 (2019)

[90] Singla, S., Feizi, S.: Salient imagenet: How to discover spurious features in deep learning? arXiv preprint arXiv:2110.04301 (2021)

[91] Singla, S., Nushi, B., Shah, S., Kamar, E., Horvitz, E.: Understanding failures of deep networks via robust feature extraction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12853–12862 (2021)

[92] Sohoni, N., Sanjabi, M., Ballas, N., Grover, A., Nie, S., Firooz, H., Ré, C.: Barack: Partially supervised group robustness with guarantees. arXiv preprint arXiv:2201.00072 (2021)

[93] Sun, C., Shrivastava, A., Singh, S., Gupta, A.: Revisiting unreasonable effectiveness of data in deep learning era. In: Proceedings of the IEEE international conference on computer vision, pp. 843–852 (2017)

[94] Tatman, R.: Gender and dialect bias in youtube's automatic captions. In: Proceedings of the first ACL workshop on ethics in natural language processing, pp. 53–59 (2017)

[95] Teney, D., Abbasnejad, E., Lucey, S., Hengel, A.v.d.: Evading the simplicity bias: Training a diverse set of models discovers solutions with superior ood generalization. arXiv preprint arXiv:2105.05612 (2021)

[96] Utama, P.A., Moosavi, N.S., Gurevych, I.: Towards debiasing nlu models from unknown biases. arXiv preprint arXiv:2009.12303 (2020)

[97] Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., SciPy 1.0 Contributors: SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods **17**, 261–272 (2020), doi:10.1038/s41592-019-0686-2

[98] Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset (2011)

[99] Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms (2017)

[100] Xiao, K., Engstrom, L., Ilyas, A., Madry, A.: Noise or signal: The role of image backgrounds in object recognition. arXiv preprint arXiv:2006.09994 (2020)

[101] Xu, Y., He, H., Shen, T., Jaakkola, T.S.: Controlling directions orthogonal to a classifier. In: International Conference on Learning Representations (2022)

[102] Yaghoobzadeh, Y., Mehri, S., Tachet, R., Hazen, T.J., Sordoni, A.: Increasing robustness to spurious correlations using forgettable examples. arXiv preprint arXiv:1911.03861 (2019)

[103] Zech, J.R., Badgeley, M.A., Liu, M., Costa, A.B., Titano, J.J., Oermann, E.K.: Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. PLoS medicine **15**(11), e1002683 (2018)

[104] Zhai, X., Puigcerver, J., Kolesnikov, A., Ruyssen, P., Riquelme, C., Lucic, M., Djolonga, J., Pinto, A.S., Neumann, M., Dosovitskiy, A., et al.: A large-scale study of representation learning with the visual task adaptation benchmark. arXiv preprint arXiv:1910.04867 (2019)

[105] Zhang, J., Marszałek, M., Lazebnik, S., Schmid, C.: Local features and kernels for classification of texture and object categories: A comprehensive study. International journal of computer vision **73**(2), 213–238 (2007)

[106] Zhang, J., Menon, A., Veit, A., Bhojanapalli, S., Kumar, S., Sra, S.: Coping with label shift via distributionally robust optimisation. arXiv preprint arXiv:2010.12230 (2020)

[107] Zhang, M., Sohoni, N.S., Zhang, H.R., Finn, C., Ré, C.: Correct-n-contrast: A contrastive approach for improving robustness to spurious correlations. In: NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications (2021)

[108] Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., Torralba, A.: Places: A 10 million image database for scene recognition. IEEE transactions on pattern analysis and machine intelligence **40**(6), 1452–1464 (2017)

Appendix Outline

This appendix is organized as follows. In Section A we present details on the experiments on feature learning in the presence of spurious correlations. In Section B we provide details on the results for the benchmark Waterbirds and CelebA datasets. We provide details on the experiments on the reliance of ImageNet-trained models on the background in Section C and on the texture-bias in Section D.

Code references. In addition to the experiment-specific packages that we discuss throughout the appendix, we used the following libraries and tools in this work: NumPy [31], SciPy [97], PyTorch [73], Jupyter notebooks [48], Matplotlib [42], Pandas [63].

A Details: Understanding Representation Learning with Spurious Correlations

Here we provide details on the experiments in Section 4.

A.1 Feature learning on Waterbirds

Inverse problem. In Table 4 we present the results on the inverted Waterbirds problem, where the goal is to predict the background type while the bird type serves as a spurious feature. The results for the inverted problem are analogous to the results for the standard Waterbirds (Table 1): models trained on the Original data learn the background features sufficiently well to predict the background type with high accuracy when the spurious foreground feature is not present, but perform poorly when presented with conflicting background and foreground features. While it is often suggested than neural networks are biased to learn the background [100], we see that in fact the network relies on the spurious foreground feature (bird) when trained to predict the background.

Data. We show examples of Original, FG-Only and BG-Only Waterbirds images in Figure 6. To generate the data, we follow the instructions at `github.com/kohpangwei/group_DRO#waterbirds`, but in addition to the Original data we save the backgrounds and foregrounds separately. Consequently, the FG-Only data contains the same exact birds images as the Original data, and the BG-Only data contains the same exact places images as the Original data. For the 100% spurious correlation strength we simply discard all the minority groups data from the Original (95% spurious correlation) training dataset. For the Balanced data, we start with the Original data with 95% spurious correlation and replaced the background in the smallest possible number of images (chosen randomly) to achieve a 50% spurious correlation strength.

Hyper-parameters. For the experiments in this section we use a ResNet-50 model pretrained on ImageNet, imported from the `torchvision` package: `torchvision.models.resnet50(pretrained=True)` [62]. We train the models for 50 epochs with SGD with a constant learning rate of 10^{-3}, momentum decay of 0.9, batch size 32 and a weight decay of 10^{-2}. We use random crops (`RandomResizedCrop(224, scale=(0.7, 1.0), ratio=(0.75, 4./3.), interpolation=2)`) and horizontal flips (`RandomHorizontalFlip()`) implemented in `torchvision.transforms` as data augmentation.

A.2 Simplicity bias

Data. We show data examples from Dominoes datasets (MNIST-MNIST, MNIST-FashionMNIST and MNIST-CIFAR) in Figure 6. The top half of each image shows MNIST digits from classes $\{0, 1\}$, and the bottom half shows: MNIST images from classes {7,

Figure 6: **Data examples.** Variations of the waterbirds (Top) and Dominoes (Bottom) datasets generated for the experiment in Section 4.

Train Data	**Spurious Corr. (%)**	**Test Data (Worst/Mean, %)**	
		Original	BG-Only
Balanced	50	93.2/95.6	93.6/96.0
Original	95	77.4/91.2	93.1/95.7
Original	100	36.1/77.5	92.7/94.8
Place-Only	-	91.8/94.2	92.4/95.2

Table 4: **Feature learning on Inverted Waterbirds.** ERM classifiers trained on Inverted Waterbirds with Original and BG-Only images. Here the target is associated with the background type, and the foreground (bird type) serves as the spurious feature. All the models trained on the Original data including the model trained without any minority group examples (*Spurious corr.* 100%) underperform on the worst-group accuracy on the Original data, but perform well on the BG-Only data, almost matching the performance of the BG-Only trained model.

9} for MNIST-MNIST, Fashion-MNIST images from classes {coat, dress} for MNIST-FashionMNIST, and CIFAR-10 images from classes {car, truck} for MNIST-CIFAR. The label corresponds to the more complex bottom part of the image, but the top and bottom parts are correlated (we consider 95%, 99% and 100% levels of spurious correlation strength in experiments). 20% of the training data was reserved for validation or *reweighting* dataset (see Section 5) where each group is equally represented.

Hyper-parameters. We used a randomly initialized ResNet-20 architecture for this set of experiments. We trained the network for 500 epochs with SGD with batch size 32, weight decay 10^{-3}, initial learning rate value 10^{-2} and a cosine annealing learning rate schedule. For the logistic regression model, we first extract the embeddings from the penultimate layer of the network, then use the logistic regression implementation (`sklearn.linear_model.LogisticRegression`) from the `scikit-learn` package [74]. We use ℓ_1 regularization and tune the inverse regularization strength parameter C in the range $\{0.1, 10, 100, 1000\}$. For more details on Deep Feature Reweighting implementation and tuning, see section B.

Transfer from simple to complex features. In addition to the results presented in Figure 2, we measure the decoded accuracy using the model trained just on the spurious

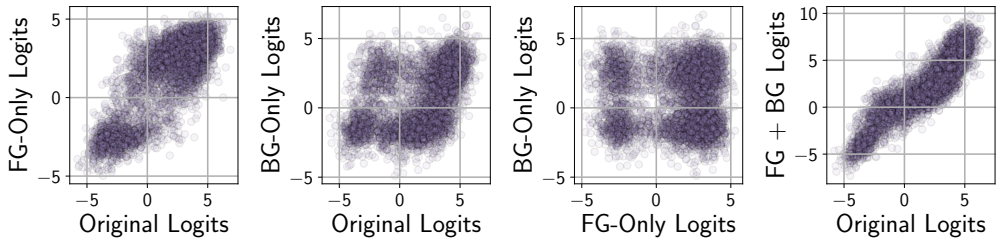


Figure 7: **Logit additivity.** Distribution of logits for the negative class on the test data for a model trained on Waterbirds dataset. We show scatter plots of the logits for the Original, FG-Only and BG-Only images. The logits on Original images are well aligned with sums of logits on the corresponding FG and BG images (rightmost panel), suggesting that the foreground and background features are processed close to independently in the network.

simple features: the top half of the image showing an MNIST digit. After training, we retrain the last layer of the model using a validation split of the corresponding Dominoes dataset which has both top and bottom parts. In this case, we measure the transfer learning performance with the features learned on the binary MNIST classification problem applied to the more complex bottom half of the image. We obtain the following *transfer accuracy* results: 92.5% on MNIST-MNIST, 92.1% on MNIST-Fashion and 61.4% on MNIST-CIFAR. On all datasets, the decoded accuracy reported in Figure 2 for the spurious correlation levels 99% and 95% is better than the transfer accuracy. For the 100% spurious correlation, transfer achieves comparable results on MNIST-Fashion and MNIST-CIFAR, but on MNIST-MNIST the decoded accuracy with a model trained on the data with the core feature is significantly higher (99% vs 92%). These results confirm that for the spurious correlation strength below 100%, the model learns a high quality representation of the core features, which cannot be explained by transfer learning from the spurious feature.

A.3 Logit additivity

To better understand why the models trained on the Original Waterbirds data perform well on FG-Only images, in Figure 7 we inspect the logits of a trained model. We show scatter plots of logits for the negative class (logits for the positive class behave analogously) on the Original, FG-Only and BG-Only test data. Both logits on FG-Only and BG-Only data correlate with the logits on the Original images, and FG-Only show a higher correlation. The FG-Only and BG-Only logits are not correlated with each other, as in test data the groups are balanced and the foreground and background are independent.

We find that the sum of the logits for the BG-Only and the logits for the FG-Only images provides a good approximation of the logits on the corresponding Original image (combining the foreground and the background). We term this phenomenon *logit additivity*: on Waterbirds, logits for the different predictive features (both core and spurious) are computed close to independently and added together in the last classification layer.

B Details: Spurious Correlation Benchmarks

Here we provide details on the experiments in Section 6.

Data. We use the standard Waterbirds and CelebA datasets, following e.g. [85, 58, 43]. See Figure 3 for group descriptions and example images.

Base model hyper-parameters. For the experiments presented in Table 6 we use a ResNet-50 model pretrained on ImageNet, imported from the `torchvision`

Method	ImageNet Pretrain	Dataset Fine-tune	Waterbirds		CelebA	
			Worst(%)	Mean(%)	Worst(%)	Mean(%)
Base Model	✓	✓	$74.9_{\pm 2.4}$	$98.1_{\pm 0.1}$	$46.9_{\pm 2.8}$	$95.3_{\pm 0}$
$\text{DFR}^{\text{Tr}}_{\text{Tr}}$	✓	✓	$90.2_{\pm 0.8}$	$97.0_{\pm 0.3}$	$80.7_{\pm 2.4}$	$85.4_{\pm 0.4}$
$\text{DFR}^{\text{Val}}_{\text{Tr}}$	✓	✓	$92.9_{\pm 0.2}$	$94.2_{\pm 0.4}$	$88.3_{\pm 1.1}$	$89.6_{\pm 0.4}$
Base Model	✗	✓	$6.9_{\pm 3.0}$	$88.0_{\pm 1.1}$	$39.8_{\pm 2.0}$	$95.7_{\pm 0.1}$
$\text{DFR}^{\text{Tr}}_{\text{Tr}}$	✗	✓	$45.4_{\pm 4.1}$	$69.8_{\pm 7.0}$	$83.4_{\pm 2.6}$	$87.5_{\pm 0.3}$
$\text{DFR}^{\text{Val}}_{\text{Tr}}$	✗	✓	$53.9_{\pm 1.8}$	$62.6_{\pm 2.2}$	$85.0_{\pm 2.1}$	$87.6_{\pm 0.3}$
$\text{DFR}^{\text{Tr}}_{\text{IN}}$	✓	✗	$47.5_{\pm 2.5}$	$52.9_{\pm 0.5}$	$77.2_{\pm 1.1}$	$83.3_{\pm 0.1}$
$\text{DFR}^{\text{Val}}_{\text{IN}}$	✓	✗	$50.5_{\pm 2.3}$	$54.4_{\pm 1.2}$	$73.1_{\pm 2.6}$	$80.9_{\pm 0.5}$

Table 5: **Effect of ImageNet pretraining and dataset fine-tuning.** Results for DFR on the Waterbirds and CelebA datasets when using an ImageNet-trained model as a feature extractor, training the feature extractor from random initialization or initializing the feature extractor with ImageNet-trained weights and fine-tuning on the target data. The models without target dataset finetuning ($\text{DFR}^{\text{Tr}}_{\text{IN}}$ and $\text{DFR}^{\text{Val}}_{\text{IN}}$) lead to relatively poor performance on both datasets. On Waterbirds, both ImageNet pretraining and dataset finetuning are needed to achieve strong performance. On CelebA, we can achieve competitive resutls training the feature extractor from a random initialization. All methods in Table 2 use ImageNet-trained models as initialization and finetune on the target dataset.

package: `torchvision.models.resnet50(pretrained=True)`. We use random crops (`RandomResizedCrop(224, scale=(0.7, 1.0), ratio=(0.75, 4./3.), interpolation=2)`) and horizontal flips (`RandomHorizontalFlip()`) implemented in `torchvision.transforms` as data augmentation. We train all models with SGD with momentum decay of 0.9 and a constant learning rate. On Waterbirds, we train the models for 100 epochs with weight decay 10^{-3}, learning rate 10^{-3} and batch size 32. On CelebA, we train the models for 50 epochs with weight decay 10^{-4}, learning rate 10^{-3} and batch size 128. We do not use early stopping.

DFR details. For DFR, we first extract and save the embeddings (inputs to the classification layer of the base model) of the training, validation and testing data using the base model. We then preprocess the embeddings to have zero mean and unit standard deviation using the standard scaler `sklearn.preprocessing.StandardScaler` from the `scikit-learn` package. In each case, we compute the preprocessing statistics on the reweighting data used to train the last layer. To retrain the last layer, we use the logistic regression implementation (`sklearn.linear_model.LogisticRegression`) from the `scikit-learn` package. We use ℓ_1 regularization. For $\text{DFR}^{\text{Val}}_{\text{Tr}}$ and $\text{DFR}^{\text{Tr}}_{\text{Tr-NM}}$ we only tune the inverse regularization strength parameter C: we consider the values in range $\{1., 0.7, 0.3, 0.1, 0.07, 0.03, 0.01\}$ and select the value that leads to the best worst-group performance on the available validation data (as described in Section 6, for $\text{DFR}^{\text{Val}}_{\text{Tr}}$ we use half of the validation to train the logistic regression model and the other half to tune the parameters at the tuning stage). For $\text{DFR}^{\text{Tr}}_{\text{Tr}}$ we additionally tune the class weights: we set the weight for one of the classes to 1 and consider the weights for the other class in range $\{1, 2, 3, 10, 100, 300, 1000\}$; we then switch the classes and repeat the procedure. For the final evaluation, we use the best values of the hyper-parameters obtained during the tuning phase and train a logistic regression model on all of the available reweighting data. We train the logistic regression model on the reweighting data 10 times with random subsets of the data (we take all of the data from the smallest group, and subsample the other groups randomly to have the same number of datapoints) and average the weights of the learned models. We report the model with averaged weights.

Base Model Hypers				Method	Waterbirds	
lr	wd	batch size	aug		Worst(%)	Mean(%)
10^{-3}	10^{-3}	32	✓	Base Model	$74.9_{\pm 2.4}$	$98.1_{\pm 0.1}$
				$\text{DFR}_{\text{Tr}}^{\text{Val}}$	$92.9_{\pm 0.2}$	$94.2_{\pm 0.4}$
10^{-3}	10^{-3}	32	✗	Base Model	73.4	97.7
				$\text{DFR}_{\text{Tr}}^{\text{Val}}$	90.9	92.2
10^{-3}	10^{-2}	32	✓	Base Model	24.1	94.4
				$\text{DFR}_{\text{Tr}}^{\text{Val}}$	88.0	88.6
10^{-3}	10^{-2}	32	✗	Base Model	66.4	97.1
				$\text{DFR}_{\text{Tr}}^{\text{Val}}$	90.1	90.5
$3 \cdot 10^{-3}$	10^{-3}	32	✓	Base Model	71.5	98.1
				$\text{DFR}_{\text{Tr}}^{\text{Val}}$	91.9	93.5
$3 \cdot 10^{-3}$	10^{-3}	32	✗	Base Model	76.5	98.0
				$\text{DFR}_{\text{Tr}}^{\text{Val}}$	89.5	94.5
10^{-3}	10^{-3}	64	✓	Base Model	73.5	98.0
				$\text{DFR}_{\text{Tr}}^{\text{Val}}$	93.1	95.0
10^{-3}	10^{-3}	64	✗	Base Model	69.5	97.4
				$\text{DFR}_{\text{Tr}}^{\text{Val}}$	89.0	93.6

Base Model Hypers				Method	CelebA	
lr	wd	batch size	aug		Worst(%)	Mean(%)
10^{-3}	10^{-4}	128	✓	Base Model	$46.9_{\pm 2.8}$	$95.3_{\pm 0}$
				$\text{DFR}_{\text{Tr}}^{\text{Val}}$	$88.3_{\pm 1.1}$	$91.3_{\pm 0.3}$
10^{-3}	10^{-3}	128	✓	Base Model	$44.3_{\pm 6.4}$	$95.2_{\pm 0.1}$
				$\text{DFR}_{\text{Tr}}^{\text{Val}}$	$86.2_{\pm 1.2}$	$90.8_{\pm 0.7}$
10^{-3}	10^{-4}	128	✗	Base Model	$46.7_{\pm 0.0}$	$95.3_{\pm 0.1}$
				$\text{DFR}_{\text{Tr}}^{\text{Val}}$	$86.9_{\pm 1.1}$	$91.6_{\pm 0.2}$
10^{-3}	10^{-3}	128	✗	Base Model	$40.6_{\pm 8.7}$	$95.1_{\pm 0.2}$
				$\text{DFR}_{\text{Tr}}^{\text{Val}}$	$85.6_{\pm 1.4}$	$91.8_{\pm 0.5}$

Table 6: **Effect of base model hyper-parameters.** We report the results of $\text{DFR}_{\text{Tr}}^{\text{Val}}$ for a range of base model hyper-parameters on Waterbirds and CelebA as well as the performance of the corresponding base models. While the quality of the base model has an effect on DFR, the results are fairly robust. For a subset of configurations we report the mean and standard deviation over 3 independent runs of the base model and DFR.

Do we need ImageNet pretraining? In Table 5 we report the results on Waterbirds and CelebA for models trained from random initialization, without ImageNet pretraining. While on Waterbirds pretraining is required to achieve good results with DFR, on CelebA we can achieve strong performance training from random initialization. We note that the baseline methods considered in Section 6 all use ImageNet-pretrained models.

What if we just use ImageNet features? As a baseline, we apply DFR to features extracted by a model pretrained on ImageNet with no fine-tuning on CelebA and Waterbirds data. We report the results in Table 5 ($\text{DFR}_{\text{IN}}^{\text{Tr}}$ and $\text{DFR}_{\text{IN}}^{\text{Val}}$ lines). While on CelebA it is

possible to get above random-guess performance with ImageNet features, on Waterbirds the performance is close to random guess. The results in Table 5 suggest that both ImageNet pretraining and fine-tuning on the target data are needed to train the best feature extractor for DFR.

Robustness to base model hyper-parameters In Table 6 we report the results of $\text{DFR}^{\text{Val}}_{\text{Tr}}$ for a range of configurations of the baseline model hyper-parameters. While the quality of the base model clearly has an effect on DFR performance, we achieve competitive results for all the hyper-parameter configurations that we consider, even when the base model performs poorly. For example, on Waterbirds with data augmentation, learning rate 10^{-3} and weight decay 10^{-2} the base model achieves worst group accuracy of 24.1%, but by retraining the last layer of this model with $\text{DFR}^{\text{Val}}_{\text{Tr}}$ we still achieve 88% worst group accuracy.

B.1 Prior work assumptions

In Section 6 we compare DFR to ERM, Group-DRO [85], JTT [58], SUBG [43] and SSA [68]. These methods differ in assumptions about the amount of group information available.

Group-DRO and SUBG assume that both train and validation data have group labels, and the hyper-parameters are tuned using worst-group validation accuracy. $\text{DFR}^{\text{Tr}}_{\text{Tr}}$ and $\text{DFR}^{\text{Tr}}_{\text{Tr-NM}}$ match the setting of these methods and use the same data and group information; in particular, these methods only use the validation set with group labels to tune the hyper-parameters.

A number of prior works [e.g. 58, 18] sidestep the assumption of knowing the group labels on train data, but still rely on tuning hyper-parameters using worst-group accuracy on validation, and thus, having group labels on validation data. In fact, Idrissi et al. [43] showed that ERM is a strong baseline when tuned with worst-group accuracy on validation.

Lee et al. [56] explore the setting where they do not necessarily require group labels on validation data, but their method implicitly relies on the presence of sufficiently many minority examples in the validation set such that different prediction heads would disagree on those examples to choose the most reliable classifier.

Recent works Nam et al. [68] and Sohoni et al. [92] explore the setting where the group information is available on a small subset of the data (e.g. on the validation set), and the goal is to use the available data optimally, both to train the model and to tune the hyper-parameters. These methods use semi-supervised learning to extrapolate the available group labels to the rest of the training data. We consider this same setting with $\text{DFR}^{\text{Val}}_{\text{Tr}}$, where we use the validation data to retrain the last layer of the model.

C Details: ImageNet Background Reliance

Here we provide details on the experiments on background reliance in Section 7.

Data. We use the ImageNet-9 dataset [100]. To test whether our models can generalize to unusual backgrounds, we additionally generate *Paintings-BG* data shown in Figure 9. For the Paintings-BG data we use the Original images and segmentation masks for ImageNet-9 data provided by Xiao et al. [100], and combine them with random paintings from Kaggle's `painter-by-numbers` dataset available at kaggle.com/c/painter-by-numbers/ as backgrounds. For the ImageNet-R dataset [35], we only use the images that fall into one of the ImageNet-9 categories, and evaluate the accuracy with respect to these categories. We show examples of images from different dataset variations in Figure 9.

Base model hyper-parameters. We use a ResNet-50 model pretrained on ImageNet and a VIT-B-16 model pretrained on ImageNet-21k and fine-

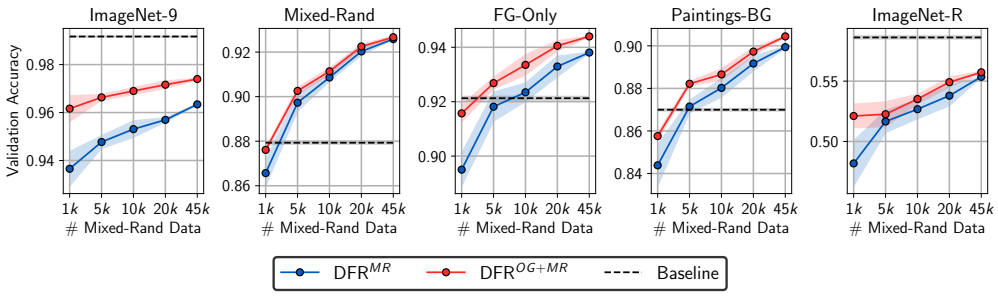
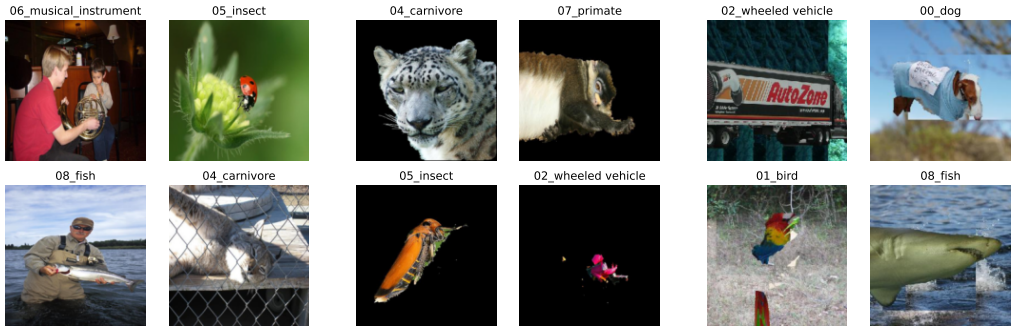


Figure 8: **ImageNet background reliance (VIT-B-16).** Performance of DFR trained on MixedRand data and MixedRand + Original data on different ImageNet-9 validation splits. All methods use an VIT-B-16 feature extractor trained on ImageNet21k and finetuned on ImageNet. DFR can reduce background reliance with a minimal drop in performance on the Original data. See Figure 5 for analogous results with a ResNet-50 feature extractor.

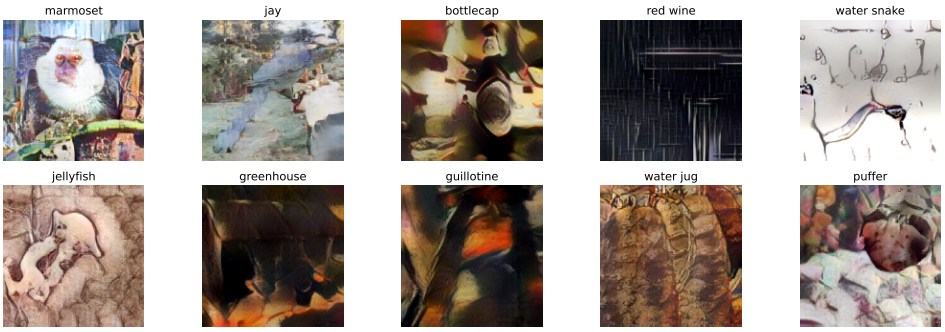
tuned on ImageNet. The ResNet-50 model is imported from the `torchvision` package: `torchvision.models.resnet50(pretrained=True)`. The VIT-B-16 model is imported from the `lukemelas/PyTorch-Pretrained-ViT` package available at github.com/lukemelas/PyTorch-Pretrained-ViT; we use the command: `ViT('B_16_imagenet1k', pretrained=True)` to load the model. To extract the embeddings, we remove the last linear classification layer from each of the models. We preprocess the data using the `torchvision.transforms` paclage `Compose([ Resize(resize_size), CenterCrop(crop_size), ToTensor(), Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])])`, where `(resize_size, crop_size)` are equal to (256, 224) for the ResNet-50 and (384, 384) for the VIT-B-16. We do not apply any data augmentation.

DFR details. We Train DFR on random subsets of the Mixed-Rand train data of different sizes (DFR$^{\text{MR}}$) or combinations of the Mixed-Rand and Original data (DFR$^{\text{OG+MR}}$). For DFR$^{\text{OG+MR}}$ we use the same number of Original and Mixed-Rand datapoints in all experiments. As the full ImageNet-9 contains $50k$ datapoints, we train the logistic regression on GPU with a simple implementation in PyTorch [73]. We then preprocess the embeddings to have zero mean and unit standard deviation using the standard scaler `sklearn.preprocessing.StandardScaler` from the `scikit-learn` package. In each case, we compute the preprocessing statistics on the reweighting data used to train the last layer. For the experiments in this section we use ℓ_2 regularization (as the number of datapoints is large relative to the number of observations, we do not have to use ℓ_1). We set the regularization coefficient λ to be 100 and train the logistic regression model with the loss $\sum_{x,y\in\mathcal{D}'} L(y, wx+b) + \frac{\lambda}{2}\|w\|^2$, where $L(\cdot,\cdot)$ is the cross-entropy loss. We use full-batch SGD with learning rate 1 and no momentum to train the model for 1000 epochs. We did not tune the λ parameter or SGD hyper-parameters.

Results for VIT-B-16. We report the results for the VIT-B-16 base model in Figure 8. The results are generally analogous to the results for ResNet-50: it is possible to significantly reduce the reliance of the model on the background by retraining the last layer with DFR. For the VIT, removing the background dependence hurts the performance on the Original data slightly, but greatly improves the performance on the images with unusual backgrounds (Mixed-Rand, FG-Only, Paintings-BG). Removing the background dependence does not improve the performance on ImageNet-R.

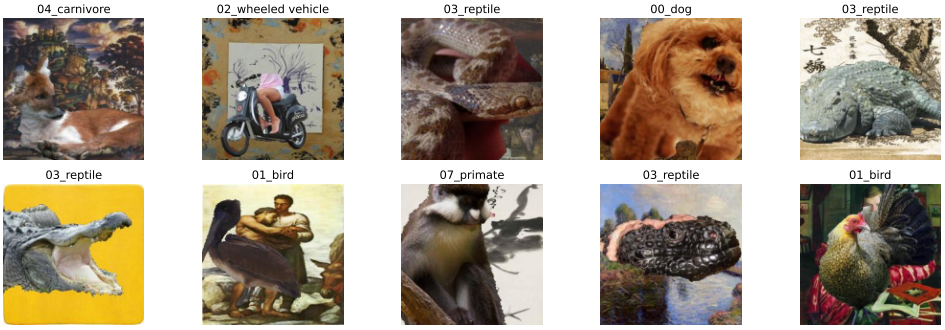


(a) IN-9 Original [100] (b) IN-9 FG-Only [100] (c) IN-9 Mixed-Rand [100]



(d) Stylized ImageNet [27]

(e) ImageNet-R [35]



(f) ImageNet-9, Paintings-BG (ours)

Figure 9: **ImageNet variations.** Examples of datapoints from the ImageNet variations used in the experiments.

D Details: ImageNet Texture Bias Details

Here we provide details on the experiments on texture bias in Section 7.

Data. We generate the stylized ImageNet (SIN) data following the instructions at github.com/rgeirhos/Stylized-ImageNet [27]. For evaluation, we use ImageNet-C [36] and ImageNet-R datasets [35]. For ImageNet-C we report the average performance across all 19 corruption types and 5 corruption intensities. We show examples of stylized ImageNet images in Figure 9.

Base model hyper-parameters. We use the same ResNet-50 and VIT-B-16 models as described in Appendix C.

DFR details. We train DFR using the embeddings of the original ImageNet (IN), stylized ImageNet (SIN) and their combination (IN+SIN) as the reweighting dataset. We preprocess the base model embeddings by manually subtracting the mean and dividing by standard deviation computed on the reweighting data used to train DFR; we did not use `sklearn.preprocessing.StandardScaler` due to the large size of the datasets ($1.2M$ datapoints for IN and SIN; $2.4M$ datapoints for IN+SIN). We train the logistic regression for the last layer on a single GPU with a simple implementation in PyTorch. We use SGD with learning rate 1, no momentum, no regularization and batch size of 10^4 to train the model for 100 epochs. We tuned the batch size (in the range $\{10^3, 10^4, 10^5\}$) and picked the batch size that leads to the best performance on the ImageNet validation set (10^4). We did not tune the other hyper-parameters.

Results for VIT-B-16. We report the results for the VIT-B-16 base model in Table 7. For this model, baselines trained from scratch on IN+SIN and SIN are not available so we only report the results for the standard model trained on ImageNet21k and finetuned on ImageNet; for DFR we report the results using IN+SIN as the reweighting dataset. Despite the large-scale pretraining on ImageNet21k, we find that we can still improve the shape bias (36% $\rightarrow$ 39.9%) as well as robustness to ImageNet-C corruptions (49.7% $\rightarrow$ 52% Top-1 accuracy). On the original ImageNet and ImageNet-R the performance of DFR is similar to that of the baseline model.

Detailed texture bias evaluation. To provide further insight into the texture bias of the models trained with DFR, in Figure 10 we report the fraction of shape and texture decisions for different classes following Geirhos et al. [27]. We produce the figure using the `modelvshuman` codebase available at github.com/bethgelab/model-vs-human [26]. We report the results for both DFR models and models trained from scratch on IN, SIN and IN+SIN as well as the ShapeResNet-50 model and humans (results from Geirhos et al. [27]). When trained on the same data, models trained from scratch achieve a higher shape bias than DFR models, but DFR can still significantly improve the shape bias compared to the base model trained on IN.

Detailed ImageNet-C results. In Figure 11, we report the Top-1 accuracy for DFR models and models trained from scratch on IN, SIN and IN+SIN on individual ImageNet-C datasets. We use the ResNet-50 base model. The model trained from scratch on IN+SIN provides the best robustness across the board, but DFR trained on IN+SIN also provides an improvement over the baseline RN50(IN) model on many corruptions.

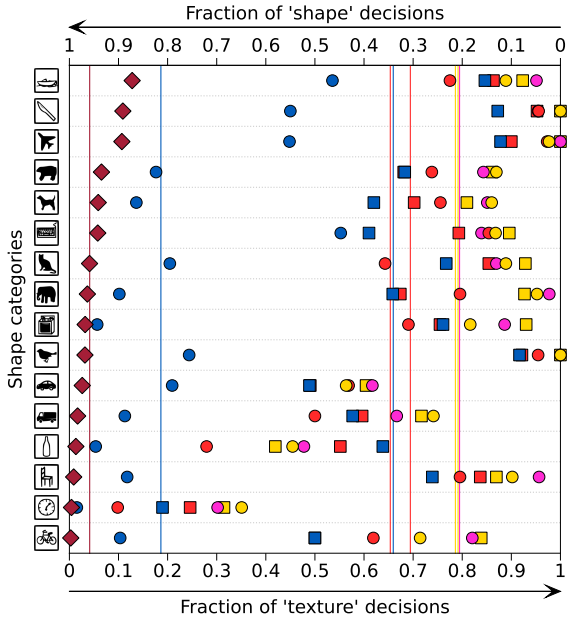


Figure 10: **Shape-texture bias report.** Detailed report of the shape-texture bias generated using the `model-vs-human` codebase (`https://github.com/bethgelab/model-vs-human`) [26]. The plot shows the fraction of the decisions made based on shape and texture information respectively on examples with conflicting cues [27]. The brown diamonds (⋄) show human predictions and the circles (○) show the performance of ResNet-50 models trained on different datasets: ImageNet (IN, yellow), Stylized ImageNet (SIN, blue), ImageNet + Stylized ImageNet (IN+SIN, red), ImageNet + Stylized ImageNet Finetuned on ImageNet (IN+SIN→IN, pink); these models are provided in the `model-vs-human` codebase. For each dataset (except for IN+SIN→IN) we report the results for DFR using an ImageNet-trained ResNet-50 model as a feature extractor with squares □ of the corresponding colors. Reweighting the features in a pretrained model with DFR we can significatly increase the shape bias: DFR trained on SIN (blue squares) virtually matches the shape bias of the model trained from scratch on IN+SIN (red circles). However, the model trained just on SIN (blue circles) from scratch still provides a significantly higher shape bias, that we cannot match with DFR.

Method	**Training Data**	**Shape bias (%)**	**Top-1 Acc (%) / Top-5 Acc (%)**		
			ImageNet	ImageNet-R	ImageNet-C
VIT-B-16	IN21k + IN	36	79.2/95.0	29.1/42.0	49.7/69.3
DFR	IN+SIN	39.9	79.7/94.5	29.0/41.4	52.0/71.0

Table 7: **Texture-vs-shape bias results for VIT-B-16.** Shape bias, top-1 and top-5 accuracy on ImageNet validation set variations for VIT-B-16 pretrained on ImageNet21k and finetuned on ImageNet and DFR using this model as a feature extractor and reweighting the features on combined ImageNet and Stylized ImageNet datasets. By retraining just the last layer with DFR, we can increase the shape bias compared to the feature extractor model and improve robustness to covariate shift on ImageNet-C.

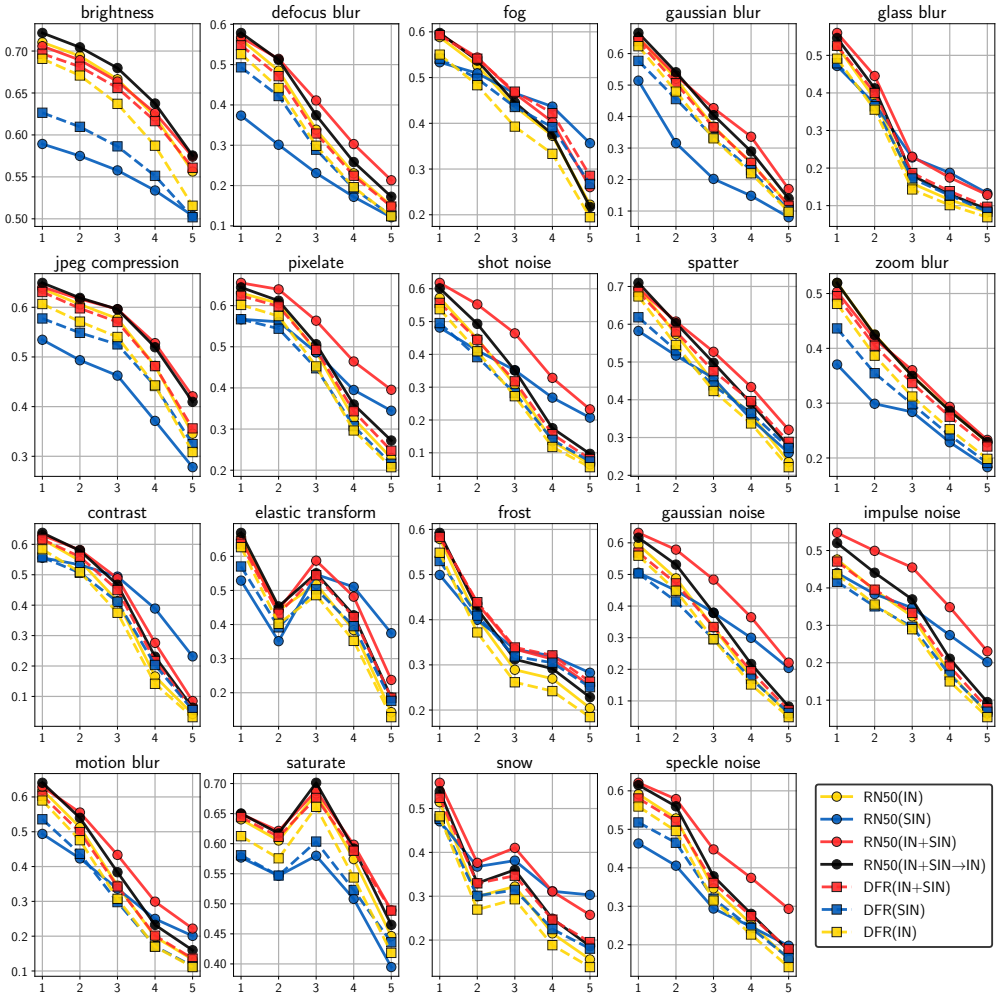


Figure 11: **ImageNet-C results.** Results of ResNet-50 models trained on different ImageNet variations (shown in circles) and DFR using an ImageNet-trained ResNet-50 model as a feature extractor on ImageNet-C and ImageNet-R datasets. Each panel corresponds to a different corruption, and the horizontal axis represents the corruption intensity. Retraining the last layer on ImageNet-Stylized (DFR(SIN), red squares) improves robustness to ImageNet-C corruptions compared to the base model (RN50(IN), blue circles), but does not match the robustness of a model trained from scratch on SIN or IN+SIN.