computer sciences and mathematics forum

MDPI

Proceeding Paper

Frustratingly Easy Environment Discovery for Invariant Learning †

Samira Zare * and Hien Van Nguyen

Electrical and Computer Engineering Department, University of Houston, Houston, TX 77204, USA; hvnguy35@central.uh.edu

* Correspondence: szare836@cougarnet.uh.edu

† Presented at the 2nd AAAI Workshop on Artificial Intelligence with Biased or Scarce Data (AIBSD), Vancouver, BC, Canada, 26 February 2024.

Abstract: Standard training via empirical risk minimization may result in making predictions that overly rely on spurious correlations. This can degrade the generalization to out-of-distribution settings where these correlations no longer hold. Invariant learning has been shown to be a promising approach for identifying predictors that ignore spurious correlations. However, an important limitation of this approach is that it assumes access to different "environments" (also known as domains), which may not always be available. This paper proposes a simple yet effective strategy for discovering maximally informative environments from a single dataset. Our frustratingly easy environment discovery (FEED) approach trains a biased reference classifier using a generalized cross-entropy loss function and partitions the dataset based on its performance. These environments can be used with various invariant learning algorithms, including Invariant Risk Minimization, Risk Extrapolation, and Group Distributionally Robust Optimization. The results indicate that FEED can discover environments with a higher group sufficiency gap compared to the state-of-the-art environment inference baseline and leads to improved test accuracy on CMNIST, Waterbirds, and CelebA datasets.

Keywords: environment discovery; invariant learning; out-of-distribution generalization; biased data

Citation: Zare, S.; Nguyen, H.V. Frustratingly Easy Environment Discovery for Invariant Learning. *Comput. Sci. Math. Forum* **2024**, *9*, 2. https://doi.org/10.3390/cmsf2024009002

Academic Editors: Kuan-Chuan Peng, Abhishek Aich and Ziyan Wu

Published: 29 January 2024

1. Introduction

Artificial intelligence systems may exhibit bias stemming from data (data bias), and algorithmic design choices can expedite erroneous decisions during training (algorithm bias) [1,2]. Generally, neural networks often exploit spurious correlations in training data as shortcuts to make predictions [1,3,4]. This leads to suboptimal performance on examples where the learned shortcuts do not apply [5–7]. This performance gap is observed across various applications like medical imaging [8–10], and facial recognition [5,11]. Recent methods have sought to mitigate unintended biases in AI systems through interventions before (pre-processing), during (in-processing), or after (post-processing) training [12]. In-processing approaches directly target algorithmic design to alleviate biases by adjusting sample importance [7,13,14], employing adversarial learning [15,16], or incorporating invariant learning [3,17]. While these methods effectively address the problem, they rely on having access to diverse environments (also known as domains) or prior knowledge of protected groups. Unfortunately, obtaining such information is usually infeasible due to expensive annotations, challenges in effectively grouping datasets, and privacy and ethical constraints [18]. An approximation becomes necessary when the system does not have direct access to diverse environments or protected groups.

Our goal is to strategically partition a training dataset and estimate distinct environments (domains) to facilitate the use of invariant learning algorithms for bias removal. Invariant learning methods learn an invariant predictor that remains robust across environments [3,5,17], making them more effective compared to other debiasing approaches [3]. Similar efforts, such as EIIL [6], also explore environment estimation for

invariant learning. However, these approaches heavily rely on the assumption that biased samples are easily identified through Empirical Risk Minimization (ERM) pre-training. Real-world scenarios, on the other hand, challenge this assumption, as the ERM approach might learn a combination of biased and causal features. Our paper acknowledges that shortcuts are learned more easily due to their simplicity, which offers an opportunity for effectively partitioning the samples. To validate our intuitions, we conducted an experiment on the Colored MNIST (CMNIST) dataset [3] with the target attribute y representing "the digit smaller than five or not" and the protected attribute a representing digit color. The target label exhibits a strong correlation with digit color (with a probability of 90%). Hence, the model can easily use color as a spurious shortcut to make predictions during training. As shown in Figure 1, training an ERM classifier revealed that the loss function rapidly decreases for biased samples, while for bias-conflicting samples, it first increases and then decreases when the model starts to overfit on all training samples. Two key observations emerged. First, bias is learned faster from early epochs, suggesting a profitable opportunity for a partitioning strategy. Second, given enough training, the ERM model can overfit even on bias-conflicting samples, confirming the limitations of naïve ERM-based approaches to separating biased samples. We propose to intentionally promote the features that are learned during the early epochs of training using the Generalized Cross-Entropy (GCE) loss function [4]. This reinforcement is followed by partitioning training samples into two environments based on model performance. The discovered environments can then be used to train invariant learning algorithms. Despite its simplicity compared to more complex baselines, FEED effectively identifies environments with a high group sufficiency gap. Our contributions can be summarized as follows:

- We present a novel environment discovery approach using the Generalized Cross-Entropy (GCE) loss function, ensuring the reference classifier leverages spurious correlations. Subsequently, we partition the dataset into two distinct environments based on the performance of the reference classifier and employ invariant learning algorithms to remove biases.
- We study the environments in invariant learning from the perspective of the "Environment Invariance Constraint" (EIC), which forms the foundation for FEED.
- We introduce the Square-MNIST dataset to evaluate the ability of our model in more challenging scenarios where the true causal features (strokes) and spurious features (squares) closely resemble each other. Our evaluation demonstrates the superior performance of FEED compared to other environment discovery approaches.

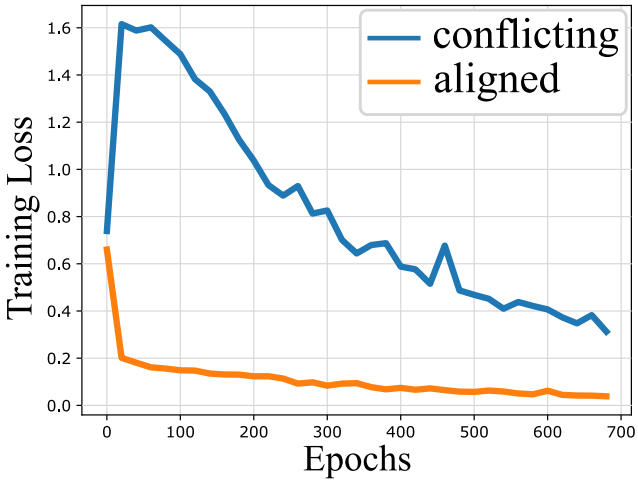

Figure 1. Training dynamics for CMNIST benchmark. For bias-aligned samples, the label y can be easily predicted based on the spurious associations, however, for other samples, this spurious correlation does not apply. While the loss for bias-aligned samples decreases quickly, for other samples the loss increases at early epochs.

2. Related Works

Bias Removal without Environment Labels. Since obtaining environments or group annotations can be costly or infeasible, various methods have been proposed to remove biases by exploiting the mistakes of an ERM model (also known as reference model). One line of work utilizes these mistakes to reweigh the data for training the primary model [7,13,19–22]. For example, [7] up-weight the error samples from the reference model or [13] determine importance weights based on the relative cross-entropy losses of the reference and primary models. These methods, however, differ from ours because instead of training a classifier with curated importance weights, we trained an invariant predictor. Another line of work leverages the mistakes to apply an invariant learning algorithm [6,23,24]. Refs. [23,24] both train a GroupDRO model by inferring subclasses from the representations learned by the reference model. The most closely related work to our paper is EIIL [6], which infers the environments for invariant learning by maximizing the regularization term of IRM. The main drawback of the above-mentioned methods is the assumption that the ERM model always learns the shortcut. This is the case in benchmarks like CMNIST, which are specifically created to frustrate ERM [25]. However, we show that these methods fail miserably on simpler tasks that do not follow the assumption. Another group of works trains a separate network to find either sample weights or environment assignment probability. Ref. [26], for instance, extends DRO using an auxiliary model to compute the importance weights. However, rather than training an online fair model for accurate predictions within a given distribution, we aim to find data partitions that allow us to employ invariant learning techniques to address distribution shifts [6]. ZIN [27] also uses an auxiliary network to learn a partition function based on IRM. This structure cannot be generalized to provide environments for other robust algorithms. Ref. [28] also proposes a framework to partition the data. However, their method is limited to the case where the input can be decomposed into invariant and variant features. Other works create domains for adversarial training [29], but we focus on invariant learning due to the limitations of adversarial methods.

Invariant Learning. Recent studies have addressed biases by learning invariances in training data. Motivated by casual discovery, IRM [3] and its variants [25,30–33] learn a representation such that the optimal classifier built on top is the same for all training environments. LISA [34] also learns invariant predictors via selective mix-up augmentation across different environments. Other methods like Fish [35], IGA [36], and Fishr [37] introduce gradient alignment constraints across training environments. Another large class of methods for generalizing beyond training data is distributionally robust optimization (DRO) [5,38–40]. REx [17] and GroupDRO [5] are notable instances of DRO methods, aiming to find a solution that performs equally well across all environments. The success of the above-mentioned methods depends on environment partitions or group annotations. However, these annotations are often unavailable or expensive in practice. Beyond the methods discussed above, adversarial training is another popular approach for learning invariant or conditionally invariant representations [15,16,29,41,42]. However, the performance of adversarial training degrades in settings where distribution shift affects the marginal distribution of labels [3,42]. Due to these limitations, recent works have focused on learning invariant predictors.

3. Frustratingly Easy Environment Discovery

In this section, we present our frustratingly easy framework (FEED) for partitioning a dataset into environments (domains) tailored for invariant learning. Our approach does not require prior knowledge of environment assignments or protected groups. Instead, we assume that the training dataset is affected by a shortcut that might be learned by the model to accurately predict outcomes for the majority of samples [3,5,6,17]. This shortcut, however, does not apply to the remaining samples, which may be either bias-conflicting or bias-irrelevant. Formally, we consider a dataset $D = \{D^e\}_{e \in \text{supp}(\mathcal{E}_{tr})}$, where $x, y, e \sim p(x, y, e)$ be observational data from multiple training environments $e \in \text{supp}(\mathcal{E}_{tr})$.

In each environment, data are generated from the same input and label spaces $\mathcal{X} \times \mathcal{Y}$ according to some distribution. The environments differ in how labels are spuriously correlated with the spurious attribute $a \in \mathcal{A}$. In an invariant learning problem, the goal is to find a predictor function $f : \mathcal{X} \to \mathcal{Y}$ that generalizes well across all possible environments in $\text{supp}(\mathcal{E}) \supseteq \text{supp}(\mathcal{E}_{tr})$. However, the required environment assignments are not always available. In this paper, we aim to create useful environments to remove shortcuts and enhance generalization. After discovering the environments, we evaluate their efficacy by measuring the sufficiency gap [6] and their practical utility in mitigating biases using invariant learning.

We begin by defining the Environment Invariance Constraint (EIC) [6]. EIC is an important condition that invariant predictors must satisfy. Assume $\mathcal{H}$ is a representation space. $\Phi : \mathcal{X} \to \mathcal{H}$ denotes the parameterized mapping or model that we optimize. We refer to $\Phi(x) \in \mathcal{H}$ as the representation of sample x. Invariant models learn a representation $\Phi(x)$ that performs simultaneously optimal for all environments; i.e., it has stable relationships with y across environments. In addition, for regular loss functions like cross-entropy and mean squared error, optimal classifiers can be expressed as conditional expectations of the output variable. Therefore, the data representation function Φ must satisfy the Environment Invariance Constraint (also known as Invariance Property), defined as:

$$\begin{aligned} \mathbb{E}[y|\Phi(x) = h, e] = \mathbb{E}[y|\Phi(x) = h, e'] \\ \forall h \in \mathcal{H} \quad \forall e, e' \in \text{supp}(\mathcal{E}) \end{aligned} \tag{1}$$

This means that invariant models learn a set of features such that the conditional distribution of outcomes given the predictor is invariant across all training environments. Our goal was to partition a training dataset into environments that could promote effective invariant learning by maximally satisfying the EIC. In other words, we sought environments so that the invariant learning method could not satisfy the EIC unless it learned invariant associations and ignored shortcuts.

Following [36], we defined the invariant set as $\mathcal{I}_{\mathcal{E}} = \{\Phi(x) : y \perp \mathcal{E} | \Phi(x)\}$. Similarly, given training environments, we can define $\mathcal{I}_{\mathcal{E}_{tr}} = \{\Phi(x) : y \perp \mathcal{E}_{tr} | \Phi(x)\}$. $\mathcal{I}_{\mathcal{E}}$ is the set of features that are invariant for all possible unseen environments $e \in \text{supp}(\mathcal{E})$. However, using the training environments $\mathcal{E}_{tr}$, we can only learn $\mathcal{I}_{\mathcal{E}_{tr}}$. The learned predictor is only invariant to such limited environments, and it is not guaranteed to be invariant with respect to all possible environments $\mathcal{E}$ [28]. As a result, for a set of training environments $\mathcal{E}_{tr}$, we have $\mathcal{I}_{\mathcal{E}} \subseteq \mathcal{I}_{\mathcal{E}_{tr}}$. Intuitively, the invariant set $\mathcal{I}_{\mathcal{E}}$ is smaller because it has to generalize across all domains. Hence, not all environments are helpful to tighten the invariant set, and even available labeled environments may be insufficient for learning the optimal $\mathcal{I}_{\mathcal{E}}$, as we will empirically demonstrate in the Experiment Section. Additionally, in many real-world applications, environments may not be available. This motivated us to study how to exploit the latent intrinsic variations in training data to discover refined environments.

Since spurious attribute a can introduce shortcuts for labels y, it follows that there exist latent intrinsic spurious features Ψ in our input samples x, e.g., the digit color in CMNIST or the background in the Waterbirds dataset. However, these shortcuts can vary across domains and degrade the generalization. To put it formally, for a pair $(x, \Phi(x))$ satisfying EIC, there exists $\Psi(x)$ such that $\mathbb{E}[y|\Psi(x), e]$ can arbitrarily change across environments. Higher variation of Ψ among environments leads to a smaller $|\mathcal{I}_{\mathcal{E}_{tr}}|$ since more variant (unstable) features can be excluded by leveraging invariant learning algorithms, thereby bringing us closer to $\mathcal{I}_{\mathcal{E}}$. In this regard, we redefine our research question as "*how we can effectively partition the dataset into environments with significant variations in* $\mathbb{E}[y|\Psi, e]$".

While in general, we may require a large number of environments to tighten the $\mathcal{I}_{\mathcal{E}_{tr}}$, in most cases, two environments may suffice to recover invariance [3,6,17]. These are situations where the EIC cannot be satisfied for two different environments, e_1 and e_2, unless Φ extracts the causal invariance [3]. To discover such environments, one approach is to partition the dataset into two opposite environments based on the agreement between y

and the spurious attribute a. In one environment, the network can directly use the shortcut to make predictions (i.e., they agree). However, in the second environment, the association between the label and shortcut does not apply, meaning that the network has to use the shortcut but in a reverse manner (i.e., they disagree) to make correct predictions. This setup creates two environments with diverse $\mathbb{E}[y|\Psi(x), e]$ because the association between the label and the spurious attribute exhibits significant variation.

We aimed to generate two environments with opposite associations between labels and shortcuts. To achieve this, we trained a neural network, M, as a reference classifier for partitioning the dataset. We then compared the performance between model M and a dummy model $1 - M$ to separate bias-aligned and bias-conflicting samples. This way, we ensured that the two environments exhibited reverse associations. To guarantee that our reference classifier M utilized the shortcut for predictions, we intentionally forced M to make predictions based on the shortcut. Analyzing the training loss dynamics, we observed that the training loss of samples containing shortcuts reduced quickly, whereas the loss for other samples first increased and then decreased (Figure 1). Empirical evidence suggests that neural networks tend to rely on shortcuts that may exist in the dataset and memorize them during the early stages of training, as these concepts are often simpler than the main task [4,13,43]. Therefore, by deliberately reinforcing the predictions of model M in the early stages of training, we could encourage it to learn the intrinsic spurious features Ψ. We accomplished this using the Generalized Cross-Entropy (GCE) [44] loss function:

$$l_{GCE}(M(x), y) = \frac{1 - M_j(x)^q}{q} \tag{2}$$

where $M_j(x)$ is the softmax output for the j-th element of $M(x)$ corresponding to the target y, and $q \in (0, 1]$ is a hyperparameter to control the degree of amplification. Using L'Hopital's rule, GCE is equivalent to the standard Cross-Entropy (CE) when $q \to 0$ [44]. Compared to the Cross-Entropy, GCE weighs the gradient of each sample by an additional $M_j(x)^q$, i.e., $\frac{\partial l_{GCE}(M(x),y)}{\partial \theta} = M_j(x)^q \frac{\partial l_{CE}(M(x),y)}{\partial \theta}$, where θ is the model parameters. As a result, using the GCE loss, we could place more emphasis on samples for which the model has a higher confidence level (i.e., higher softmax output). Since the shortcut is easier and learned from the early epochs, we were encouraging our reference classifier to focus more on them.

Furthermore, it was crucial to ensure that as we continued the training of model M by increasing the number of epochs, the model did not overfit on bias-conflicting samples (Figure 1). This precaution was to guarantee that our reference classifier was making predictions solely based on the shortcut. In this regard, we proposed to train M only on bias-aligned samples. We began with two randomly assigned environments e_1 and e_2 (`np.random.randint` $\sim$ discrete uniform) with equal sizes. We then selected one of these two random environments, say e_1, as an initialization for the biased environment to train the reference classifier. After each training epoch, we updated both e_1 and e_2 based on a difficulty score that reflected how challenging each sample is. We chose to use the minimum of Cross-Entropy loss per sample for model M and model $1 - M$, as it would provide a continuous metric that could be easily compared. Since model M is intentionally biased, it exhibits superior performance (i.e., lower Cross-Entropy loss) on biased samples, while model $1 - M$ uses the shortcut in the opposite direction and performs better on bias-conflicting samples. Consequently, as we iteratively updated the environment partitions, e_1 progressively contained more bias-aligned samples, while e_2 comprised an increasing proportion of bias-conflicting samples. This approach ensures that model M continued training on an increasingly biased dataset without overfitting on all samples. Algorithm 1 provides the pseudocode for FEED. Following the partitioning of the training data into two environments, we could then apply invariant learning algorithms. Additionally, we empirically observed that we could use FEED to estimate groups based on the pair (e, y) (rather than (a, y)) for the GroupDRO algorithm and achieve favorable performance.

Algorithm 1 FEED Algorithm

Input: dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$, model M
Output: environments e_1, e_2

Randomly initialize e_1 and e_2 using `np.random.randint`
for epochs **do**
  train M by minimizing $\mathbb{E}_{p((x,y)|e_1)}\left[l_{GCE}(M(x), y)\right]$
  for $(x_i, y_i) \in D$ **do**
    if $l_{CE}\left(M(x_i), y_i\right) < l_{CE}\left(1 - M(x_i), y_i\right)$ **then**
      Assign (x_i, y_i) to e_1
    else
      Assign (x_i, y_i) to e_2
    end if
  end for
end for
return e_1, e_2

Leveraging FEED allowed us to partition the dataset into two environments with high variation in spurious correlations. In these environments, an invariant model cannot satisfy the EIC unless it ignores the shortcut. While FEED employs the Generalized Cross-Entropy (GCE) loss to promote the learning of spurious correlations, other methods such as EIIL [6] and JTT [7] use the Cross-Entropy loss to train their reference models. However, Cross-Entropy may not always recover a biased model. Furthermore, unlike prior approaches [6,7] that utilize the entire dataset to train the reference classifier, we exclusively used e_1 for training our reference classifier. This is to prevent overfitting and focusing solely on spurious correlations. Overfitting on all training samples would make partitioning the samples impossible. Moreover, rather than defining an optimization problem for environment discovery, as seen in previous works [6], we proposed a simple yet effective approach for updating the environment assignments at each epoch. Employing an optimization problem is not easily scalable to the mini-batch training paradigm of neural networks.

4. Experiments

Here, we empirically show that FEED can significantly improve the performance of invariant learning algorithms. We compare it with EIIL [6] and study how training environments created by each method help IRM [3] and REx [17] to improve generalization. We compare the performance with ERM, CVaR DRO [45], GEORGE [24], Fish [35], Spectral Decoupling (SD) [43], CORAL [46], and two recent reweighing algorithms, namely Just Train Twice (JTT) [7] and Learning from Failure (LfF) [13]. We consider GroupDRO [5] as an upper bound since it assumes access to protected group annotations.

4.1. Dataset

We used three classification datasets for which prior works have observed poor generalization performance due to spurious correlations. Figure 2 shows details of these datasets. For CMNIST dataset, we considered the task from [3], where we had a binary classification of colored digit images of $[0, 4]$ vs. $[5, 9]$. Two training environments were provided such that the color correlated with the target label with probabilities of 0.8 and 0.9. However, in the test dataset, this correlation dropped to 0.1. Also, the target label was noisy and only matched the digit class with a probability of 0.75.

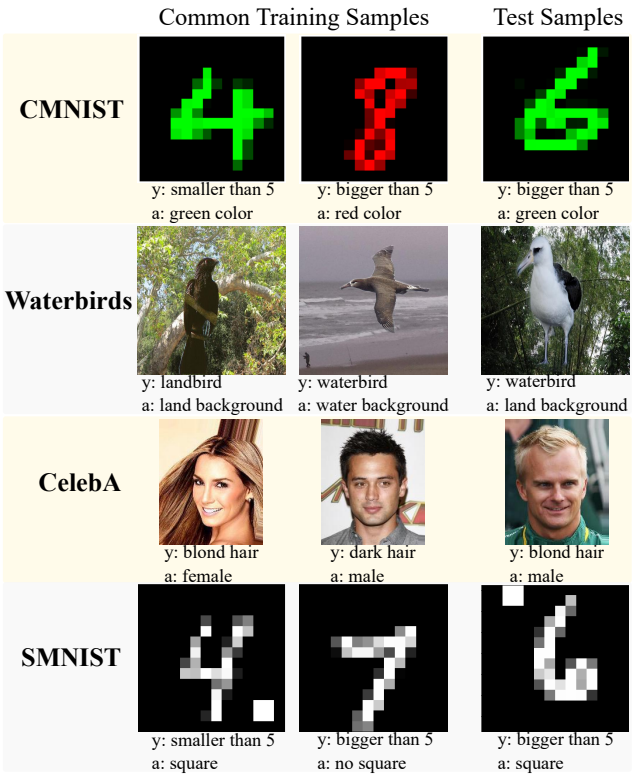

Figure 2. Sample training and test images for our datasets. The spurious correlation between the label y and the attribute a changes at test time, making the tasks challenging.

In the Waterbirds dataset, we classified waterbirds vs. landbirds [5]. The label is spuriously correlated with the image background, which is either land or water. Similarly, for the CelebA dataset, the task was to classify the hair color of celebrities as blond or not blond [5]. The label is spuriously correlated with gender, which is either male or female. We also introduced Square MNIST (SMNIST) to evaluate the effectiveness of FEED in creating useful environments when the spurious attribute created a more challenging shortcut. We used a setting similar to the standard CMNIST and create grayscale images where the spurious attribute was a square randomly placed in the corners of the image.

4.2. Implementation Details

We used an MLP architecture for CMNIST and SMNIST datasets, and a ResNet-50 for Waterbids and CelebA. For each dataset, we employed the same model architecture across all approaches. For ResNet-50, we used the PyTorch implementation with ImageNet pre-trained weights. For CMNIST, we used batch training, a learning rate of 10^{-3}, and Adam optimizer. For Waterbirds and CelebA, we used a batch size of 64 and a learning rate of 10^{-6} with SGD optimizer. For FEED, we tuned the hyperparameter q by grid searching over $q \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. This yielded $q = 0.7$ for CMNIST and Waterbirds, and $q = 0.5$ for CelebA. We repeated all experiments five times. We tuned the hyperparameters based on the "average" performance on a validation set without environment labels. All codes will be made available.

4.3. Results and Discussions

Table 1 presents average and worst-group accuracy for all approaches. The worst group refers to the group (a, y) in which the model obtains the lowest test accuracy when tuned for average validation accuracy. For Waterbirds and CelebA, IRM and REx cannot be directly applied as the environment assignments were originally unavailable. ERM achieved a good average accuracy since it minimizes the average risk over all training samples. However, its worst-group accuracy was low because there were fewer training

samples for that group. Other methods, on the other hand, attempted to have a balanced performance across different environments. Compared to the EIIL [6], using FEED to create environments for invariant learning algorithms substantially improved the worst-group accuracy. Furthermore, using FEED+(invariant learning algorithms) could outperform other approaches and achieve a comparable performance relative to GroupDRO [5]. Another interesting finding is that for CMNIST, even though the training environment labels are available, FEED can create a set of new environments (given the combination of two available environments) with which invariant learning algorithms can achieve better accuracy. Furthermore, for the CMNIST dataset, using the estimated pair of (e, y), we can fully recover (a, y). Therefore, FEED+GroupDRO will be identical to GroupDRO. In the following, we analyze the created environments and provide a probable explanation for the performance improvement.

Table 1. Test accuracy. Compared to EIIL, the environments created by FEED substantially improve worst-group accuracy. GroupDRO sets an upper bound since it assumes access to group annotations. Since environment labels for the Waterbirds and CelebA datasets are unavailable, IRM and REx are not applicable. On the CMNIST dataset, although the training environments are available, our created environments improved the performance. Experiments were repeated five times.

	CMNIST		**WaterBirds**		**CelebA**	
	Avg. Acc.	**Worst-Group Acc.**	**Avg. Acc.**	**Worst-Group Acc.**	**Avg. Acc.**	**Worst-Group Acc.**
ERM	$17.1 \pm 0.4\%$	$8.9 \pm 1.8\%$	$97.3 \pm 0.2\%$	$60.3 \pm 1.9\%$	$95.6 \pm 0.2\%$	$47.2 \pm 3.7\%$
LfF	$42.7 \pm 0.5\%$	$33.2 \pm 2.2\%$	$91.2 \pm 0.7\%$	$78.0 \pm 2.3\%$	$85.1 \pm 0.4\%$	$72.2 \pm 1.4\%$
JTT	$16.3 \pm 0.8\%$	$12.5 \pm 2.4\%$	$93.3 \pm 0.7\%$	$86.7 \pm 1.2\%$	$88.0 \pm 0.3\%$	$81.1 \pm 1.7\%$
GEORGE	$12.8 \pm 2.0\%$	$9.2 \pm 3.6\%$	$95.7 \pm 0.5\%$	$76.2 \pm 2.0\%$	$94.6 \pm 0.2\%$	$54.9 \pm 1.9\%$
CVar DRO	$33.2 \pm 0.5\%$	$27.9 \pm 1.1\%$	$96.0 \pm 1.0\%$	$75.9 \pm 2.2\%$	$82.5 \pm 0.6\%$	$64.4 \pm 2.9\%$
Fish	$46.9 \pm 0.9\%$	$35.6 \pm 1.5\%$	$85.6 \pm 0.8\%$	$64.0 \pm 1.7\%$	$93.1 \pm 0.4\%$	$61.2 \pm 1.8\%$
SD	$68.4 \pm 1.1\%$	$62.3 \pm 1.4\%$	$76.8 \pm 1.3\%$	$71.8 \pm 1.8\%$	$91.6 \pm 1.3\%$	$83.2 \pm 2.0\%$
CORAL	$65.1 \pm 2.5\%$	$60.2 \pm 4.1\%$	$90.3 \pm 1.1\%$	$79.8 \pm 2.5\%$	$93.8 \pm 0.9\%$	$76.9 \pm 3.6\%$
IRM	$66.9 \pm 1.1\%$	$58.1 \pm 3.7\%$	–	–	–	–
vREx	$68.7 \pm 0.7\%$	$63.8 \pm 2.8\%$	–	–	–	–
EIIL+IRM	$68.4 \pm 0.8\%$	$16.7 \pm 14.2\%$	$90.3 \pm 0.2\%$	$63.1 \pm 1.0\%$	$72.5 \pm 0.1\%$	$54.0 \pm 0.8\%$
EIIL+vREx	$57.4 \pm 0.8\%$	$14.7 \pm 11.8\%$	$89.7 \pm 0.8\%$	$65.2 \pm 3.4\%$	$76.4 \pm 0.7\%$	$54.9 \pm 2.6\%$
EIIL+GroupDRO	$44.4 \pm 1.0\%$	$35.2 \pm 8.2\%$	$96.9 \pm 0.8\%$	$78.7 \pm 1.0\%$	$90.7 \pm 0.5\%$	$71.3 \pm 0.9\%$
FEED+IRM (ours)	$70.4 \pm 0.02\%$	$69.7 \pm 0.8\%$	$92.3 \pm 0.2\%$	$88.4 \pm 0.9\%$	$86.0 \pm 0.5\%$	$81.3 \pm 1.4\%$
FEED+vREx (ours)	$71.1 \pm 0.08\%$	$69.1 \pm 1.2\%$	$93.3 \pm 0.3\%$	$88.6 \pm 1.0\%$	$86.9 \pm 0.8\%$	$83.7 \pm 1.4\%$
FEED+GroupDRO (ours)	$71.4 \pm 0.02\%$	$71.0 \pm 0.05\%$	$90.0 \pm 0.3\%$	$88.0 \pm 1.2\%$	$87.3 \pm 0.6\%$	$84.3 \pm 2.0\%$
GroupDRO	$71.4 \pm 0.02\%$	$71.0 \pm 0.05\%$	$93.5 \pm 0.3\%$	$91.4 \pm 1.1\%$	$92.9 \pm 0.2\%$	$88.9 \pm 2.3\%$

Analysis of Discovered Environments. We studied how samples from different groups were distributed across the environments created by FEED, as shown in Table 2. Note that we did not use such group annotations in FEED. We expected that e_1 contained samples where the shortcut exists in images. For instance, in CMNIST, we observed that e_1 only contained the samples where the label and color (spurious attribute) agree. This property is reasonable since in this dataset, the digit color and target agree for 85% of the training images (on average). All other samples were assigned to e_2 where this shortcut performed reverse and color and label disagree. Thus, $\mathbb{E}[y|\Psi, e]$ varies substantially, i.e., the correlation between color and target is unstable and varies across environments. However, the correlation between the digit shape and the target remains invariant. Consequently, when we applied an invariant learning algorithm, the model could satisfy the EIC unless it learned the digit shape. On the other hand, in the standard CMNIST training environments, there is still a slight chance of assuming an invariant association between color and target across the environments (about 10%). For the Waterbirds and CelebA, we observe similar behavior. For instance, in Waterbirds, only 56 training images from waterbirds on land

are available, out of which 50 images are assigned to e_2. We further analyzed those six images that were assigned to e_1 (shown in Figure 3). As can be seen, most of these samples have backgrounds resembling water, i.e., they are similar to waterbirds on water, which are mainly assigned to e_1 (861 vs. 195). This may explain why these six images are assigned to e_1. Note that waterbirds on the water are mostly assigned to e_1 since it is intended to contain the samples with the prevalent shortcut.

Table 2. Distribution of each group in created environments. $a = 0$ and $a = 1$ corresponds to green and red for CMNIST, land and water background in Waterbirds, and female and male for CelebA, respectively. The numbers show how each group is distributed in environments.

	CMNIST		**WaterBirds**		**CelebA**	
	e_1	e_2	e_1	e_2	e_1	e_2
$(a = 0, y = 0)$	100.0	0.0	93.8	7.2	95.5	4.5
$(a = 1, y = 0)$	0.0	100.0	16.9	83.1	99.5	0.5
$(a = 0, y = 1)$	0.0	100.0	10.7	89.3	82.8	17.2
$(a = 1, y = 1)$	100.0	0.0	81.5	18.5	30.5	69.5

Group Sufficiency Gap. Another way to explain the efficacy of our environments is by evaluating the group sufficiency gap $g = \left|\mathbb{E}[Y|\Psi(x), e_1] - \mathbb{E}[Y|\Psi(x), e_2]\right|$, defined based on the EIC [6]. This metric measures the degree to which the environment assignments can violate the EIC. We had to find a partitioning strategy that maximized g; i.e., greater g means higher variation in environments, which can lead to a tighter invariant set. In each created environment, the classifier could rely solely on the spurious attribute a to make predictions, i.e. $\Psi(x) = a$. Then the gap would be $g = \left|\mathbb{E}[Y|a, e_1] - \mathbb{E}[Y|a, e_2]\right|$. In environment e_1, all digits [5,9] ($y = 1$) are red ($a = 1$) and digits [0,4] ($y = 0$) are green ($a = 0$), while in environment e_2, all digits [5,9] ($y = 1$) are green ($a = 0$), and digits [0,4] ($y = 0$) are red ($a = 1$). In this case, we had $g = 1$, which is its maximum value. On the other hand, for the standard CMNIST environments, the gap is 0.1 [6], and for the EIIL environments $g = 0.83$. The proof is provided in the Appendixes A–C.

Figure 3. Waterbirds on land images that are assigned to e_1. In most of them, the background is or resembles water.

Why are ERM-based models not sufficient? EIIL [6] assumes that the reference model, which is trained using ERM, is learning the shortcut in the training dataset. Furthermore, recent work like JTT [7] claims to achieve out-of-distribution generalization by discovering the errors of an ERM model and then upweighing them during the next steps of training. Also, one may ask whether similar techniques like JTT, for example, can be used to partition the dataset and create environments for invariant learning methods. Although this strategy often works on datasets that were constructed to showcase out-of-distribution problems, assuming that the reference ERM model always learns the easy shortcuts is unrealistic. To il-

lustrate this claim, we constructed a variant of the CMNIST dataset where the robust feature (digit shape) was more predictive than the spurious feature (digit color) by decreasing the label noise level to 10% [25]. Table 3 compares the performance of different methods on this new dataset, called INVERSE-CMNIST. While ERM method failed on standard CMNIST (Table 1), it performed well on INVERSE-CMNIST because relying on the most predictive features (digit shape) is a good strategy for this task [25]. Other methods fail to achieve a good performance on INVERSE-CMNIST because they are based on the assumption that the ERM model is learning the shortcut. EIIL also cannot create useful environments in this case. In contrast, FEED utilizes the GCE loss function to encourage the model to learn the shortcut and increase variation among environments. In this experiment, FEED assigned environments exactly similar to the standard CMNNIST, shown in Table 2.

Table 3. Test accuracy for INVERSE-CMNIST. Although shortcut exists in the dataset, ERM can perform well. Hence, ERM-based models cannot achieve a good generalization. FEED can create effective environment partitioning that helps invariant learning algorithms.

	Avg. Acc.	**Worst-Group Acc.**
ERM	72.1%	68.1%
LfF	34.5%	13.3%
JTT	26.1%	18.1%
CVar DRO	38.4%	35.1%
SD	79.1%	74.9%
IRM	78.3%	75.3%
vREx	83.5%	81.7%
EIIL+IRM	42.8%	12.6%
EIIL+vREx	42.5%	6.0%
EIIL+GroupDRO	12.9%	2.5%
FEED+IRM (ours)	85.6%	84.7%
FEED+vREx (ours)	85.9%	85.2%
FEED+GroupDRO (ours)	86.1%	85.4%
GroupDRO	86.1%	85.4%

Figure 4 also compares the results of this experiment for EIIL and FEED for different levels of label noise. EIIL generalizes better with sufficiently high label noise (greater than 25%) but poorly with low label noise [6]. This controlled study highlights the limitations of EIIL and other ERM-based models in finding environments that emphasize the right invariances. They leave the question of how to effectively choose a reference ERM model in general open. However, FEED encourages learning the shortcut and shows promise in addressing this challenge.

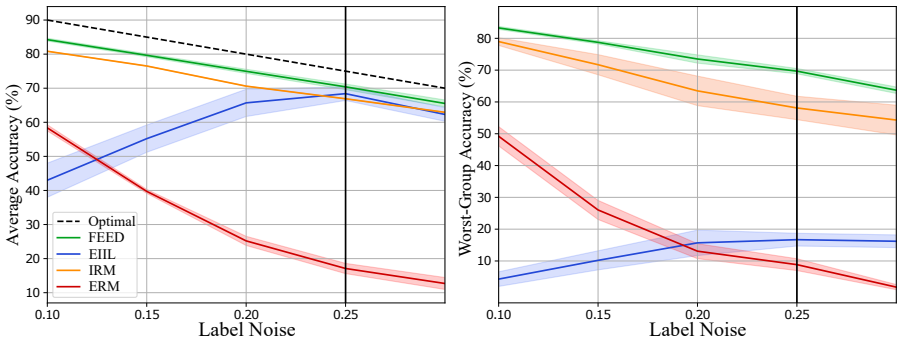

Figure 4. Test Accuracy for CMNIST with varying levels of label noise. While EIIL can only perform well under high noise, FEED consistently performs well. After both EIIL and FEED, IRM is used as the invariant learning algorithm.

What if we have more challenging shortcuts? In the introduced SMNIST dataset, although the square can serve as a shortcut, it is not as straightforward as using the color in CMNIST or the background in Waterbirds. This is because, in the feature space, the square is at a similar level as the digit strokes, making it more challenging to distinguish from the digits themselves. The results are shown in Table 4. The results indicate that the ERM model learns a mix of the shortcut and main task. Therefore, similar to the INVERSE-CMNIST, ERM-based models like EIIL cannot perform well in this task. In contrast, FEED can effectively create useful environments, although this is a challenging scenario for FEED as well. We started updating environments after a few epochs (five epochs) of training with the initial random assignments in order to give the model enough time to learn the challenging shortcut before updating the partitioning. By repeating the partitioning experiment 10 times, the group sufficiency gap for the environments created by FEED was $g = 0.98$ on average, while for EIIL, it was $g = 0.74$. Additionally, our created environments improve the performance of the invariant learning algorithms. This challenging dataset also sheds light on the effect of GCE in FEED. We repeated this experiment by replacing the GCE with standard cross-entropy (CE), as shown in Table 4. In this case, CE FEED was unable to identify the shortcut and partitioned the dataset based on the target. Therefore, invariant algorithms cannot learn a tight invariant set ($g = 0.5$).

Table 4. Test accuracy for SquareMNIST. Environments created by FEED enhance the accuracy of the invariant learning algorithms. Also, using GCE in FEED helps it find the shortcut in order to effectively partition the dataset.

	Avg. Acc.	**Worst-Group Acc.**
ERM	37.3%	4.7%
LfF	46.1%	43.3%
JTT	47.6%	23.3%
CVar DRO	49.0%	40.1%
IRM	41.6%	5.8%
vREx	54.7%	12.2%
EIIL+IRM	57.0%	32.0%
EIIL+vREx	58.9%	38.2%
EIIL+GroupDRO	67.6%	57.0%
CE FEED+IRM	36.8%	3.7%
CE FEED+vREx	33.7%	3.2%
CE FEED+GroupDRO	35.6%	8.8%
FEED+IRM (ours)	69.8%	65.0%
FEED+vREx (ours)	69.2%	63.4%
FEED+GroupDRO (ours)	71.3%	65.0%
GroupDRO	70.5%	67.8%

5. Conclusions

In this work, we presented FEED, an algorithm to create environments for invariant learning out of a biased training dataset. We provided a deep understanding of the properties of environments for invariant learning and developed FEED based on that. Specifically, we amplified the spurious attribute during training and partitioned the dataset accordingly. Without access to environment labels, FEED can outperform the invariant learning methods that require environment labels or group annotations.

Author Contributions: S.Z. is the principal author, having significantly contributed to the development of the idea, theoretical analysis, evaluations, and the composition of the paper. H.V.N. served as the advisor, playing a crucial role in validating the idea and experiments, thoroughly reviewing and revising the manuscript, and offering essential support for the research. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Cancer Institute (1R01CA277739).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The CMNIST dataset can be created using the GitHub repository https://github.com/facebookresearch/InvariantRiskMinimization (accessed on 23 January 2024). Furthermore, the Waterbirds https://nlp.stanford.edu/data/dro/waterbird_complete95_forest2water2.tar.gz (accessed on 23 January 2024) dataset and CelebA https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html (accessed on 23 January 2024) dataset are publicly available.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Loss Dynamics for CMNIST

Here, following Nam et al. [13], we study the dynamics of training loss for samples where the shortcut exists and for samples where the association between the shortcut and the label is reversed. For instance, in the CMNIST dataset, after combining, for 85% of images in class $y = 0$ (i.e., digits [0,4]) the color is green and for 85% of images in class $y = 1$ (i.e., digits [5,9]) the color is red. However, for the remaining 15% of images, this spurious association is reversed. As can be seen in Figure A1a, in this case, training loss trajectories are different for samples with and without the shortcut, especially during the early steps of training. For samples with the shortcut present, the loss quickly reduces to zero, while the loss of other samples first increases and then starts decreasing. Also, the training loss of samples where the shortcut is absent is higher, and this difference is more significant during the early stages [13]. It can also be explained using the gradient starvation concept [43], which arises when the cross-entropy loss is minimized by capturing only a subset of features relevant to the task, despite the presence of other predictive features that fail to be discovered. Neural networks have empirically been shown to prefer easy concepts during training, e.g., making predictions based on spurious shortcuts that may exist in the data [13]. Therefore, since the shortcut is often easier to learn compared to the original task (i.e., predicting based on true invariant features), neural networks tend to memorize them first. According to this observation, we developed the FEED algorithm which intentionally reinforces the predictions from the early steps of training (using the GCE loss) to make the model learn the intrinsic spurious attribute Ψ.

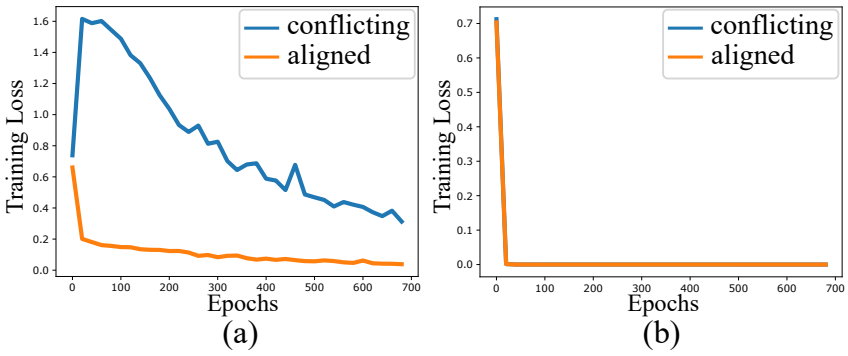

Figure A1. (**a**) Training dynamics for the standard CMNIST benchmark. For bias-aligned samples, the label y can be (easily) predicted based on the spurious associations that are prevalent in the training dataset. For other samples, this spurious correlation is reversed. While the loss for bias-aligned samples decreases quickly, for other samples the loss goes up at early epochs. (**b**) Training dynamics for predicting the color in the CMNIST dataset. We used color as the training target and digit shapes are considered as spurious attributes. Therefore, the original task is easier to learn and the loss dynamics for all samples are similar (we used batch training).

In another experiment, we study the loss dynamics for a dataset where the main task is much easier compared to the shortcut that may exist in the dataset. Specifically, we trained a model to predict the color of digits while the digit number can be a shortcut. Here, learning the original task is easy compared to the spurious attribute, and the loss

behaves similarly for all samples, as can be seen in Figure A1b. This means that the model is learning the true original task. Note that we use batch training in this experiment.

Appendix B. Using Accuracy as Difficulty Score

In this paper, we evaluate the difficulty of each sample and update the environment by computing the cross-entropy loss for both models, M and $1 - M$. While one may consider using accuracy as a difficulty score and updating the environment assignments based on correctly and incorrectly classified samples, this could lead to ambiguity. There may be cases where both models predict the output correctly or incorrectly, and it would be difficult to decide how to assign such samples. This issue is particularly prevalent during earlier epochs when the models are still learning and producing random outputs. On the other hand, using the cross-entropy loss function provides a continuous metric for measuring difficulty. We can easily update the environment assignments by comparing the loss values for each model. Thus, we choose to use the cross-entropy loss as our measure of difficulty for each sample.

Appendix C. Group Sufficiency Gap

Creager et al. [6] defined the group sufficiency gap to quantify the degree to which the EIC holds:

$$g = \left|\mathbb{E}[Y|\Psi(x), e_1] - \mathbb{E}[Y|\Psi(x), e_2]\right|$$

We can use this metric to evaluate how much a particular environment assignment can create variations w.r.t the spurious attribute across the environments. A higher group sufficiency gap is equivalent to higher variations and therefore, more variant and unstable features can be eliminated by the invariant learning algorithms. The maximum possible value for g is 1 [6]. In order to compute the group sufficiency gap, we assume the classifier is making predictions based on the spurious attribute in each environment [6].

CMNIST Standard Environments: For the standard environment assignment in the CMNIST benchmark, in environment e_1, 90% of digits [5,9] ($y = 1$) are red ($a = 1$) and 90% of digits [0,4] ($y = 0$) are green ($a = 0$), while in environment e_2, this correlation is 80%. Table A1 shows this distribution more formally. In this case, we can compute the gap as follows:

$$\begin{aligned} g &= \mathbb{E}\left|\mathbb{E}[Y|\Psi(x), e_1] - \mathbb{E}[Y|\Psi(x), e_2]\right| = \mathbb{E}\left|\mathbb{E}[Y|a, e_1] - \mathbb{E}[Y|a, e_2]\right| \\ &= \frac{1}{2}\left|\mathbb{E}[Y|a=0, e_1] - \mathbb{E}[Y|a=0, e_2]\right| + \frac{1}{2}\left|\mathbb{E}[Y|a=1, e_1] - \mathbb{E}[Y|a=1, e_2]\right| \\ &= \frac{1}{2}\left(\left|0.9(0) - 0.8(0)\right| + \left|0.1(1) - 0.2(1)\right|\right) + \frac{1}{2}\left(\left|0.1(0) - 0.2(0)\right| + \left|0.9(1) - 0.8(1)\right|\right) = 0.1 \end{aligned}$$

Therefore, the given environment splits are suboptimal w.r.t. group sufficiency gap [6], which motivates the discovery of environments with a higher group sufficiency gap.

Table A1. Distribution of each class in created environments. $a = 0$ and $a = 1$ corresponds to green and red. The numbers show the composition of samples for each class within the environments. Note that this table is different from Table 2.

	Standard CMNIST		**CMNIST FEED**		**CMNIST EIIL**		**SMNIST FEED**		**SMNIST EIIL**	
	e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2
$(a = 0, y = 0)$	90.0	80.0	100.0	0.0	93.0	7.0	99.93	4.95	85.16	9.68
$(a = 1, y = 0)$	10.0	20.0	0.0	100.0	7.0	93.0	0.07	95.05	14.84	90.32
$(a = 0, y = 1)$	10.0	20.0	0.0	100.0	6.0	89.0	0.31	98.64	15.15	89.63
$(a = 1, y = 1)$	90.0	80.0	100.0	0.0	94.0	11.0	99.69	1.36	84.85	10.37

FEED for CMNIST: As shown in Tables 2 and A1, FEED can split the CMNIST dataset based on the spurious attribute. It discovers an environment assignment based on the agreement between the label y and spurious attribute a. Therefore, the group sufficiency gap would be:

$$\begin{aligned} g &= \mathbb{E}\big|\mathbb{E}[Y|\Psi(x), e_1] - \mathbb{E}[Y|\Psi(x), e_2]\big| = \mathbb{E}\big|\mathbb{E}[Y|a, e_1] - \mathbb{E}[Y|a, e_2]\big| \\ &= \frac{1}{2}\big|\mathbb{E}[Y|a=0, e_1] - \mathbb{E}[Y|a=0, e_2]\big| + \frac{1}{2}\big|\mathbb{E}[Y|a=1, e_1] - \mathbb{E}[Y|a=1, e_2]\big| \\ &= \frac{1}{2}\big(\big|1(0) - 0(0)\big| + \big|0(1) - 1(1)\big|\big) + \frac{1}{2}\big(\big|0(0) - 1(0)\big| + \big|1(1) - 0(1)\big|\big) = 1 \end{aligned}$$

This shows that our discovered environment can achieve the maximum group sufficiency gap. We note that there can be other splitting strategies that can maximize the gap g, however, we found the proposed strategy straightforward and effective. For the INVERSE-CMNIST experiment, FEED can achieve exactly the same environment assignment. Therefore, we do not list that, separately.

EIIL for CMNIST: The distribution of each class in the environments that EIIL [6] creates, is shown in Table A1. We can find the group sufficiency gap as follows:

$$\begin{aligned} g &= \mathbb{E}\big|\mathbb{E}[Y|\Psi(x), e_1] - \mathbb{E}[Y|\Psi(x), e_2]\big| = \mathbb{E}\big|\mathbb{E}[Y|a, e_1] - \mathbb{E}[Y|a, e_2]\big| \\ &= \frac{1}{2}\big|\mathbb{E}[Y|a=0, e_1] - \mathbb{E}[Y|a=0, e_2]\big| + \frac{1}{2}\big|\mathbb{E}[Y|a=1, e_1] - \mathbb{E}[Y|a=1, e_2]\big| \\ &= \frac{1}{2}\big(\big|0.93(0) - 0.07(0)\big| + \big|0.06(1) - 0.89(1)\big|\big) + \frac{1}{2}\big(\big|0.07(0) - 0.93(0)\big| + \big|0.94(1) - 0.11(1)\big|\big) = 0.83 \end{aligned}$$

FEED for SquareMNIST: We repeated this experiment 10 times and then took the average of the environment assignments. This is to make sure our results are stable and reproducible. The distribution for each environment is given in Table 2. In this case, the group sufficiency gap would be:

$$\begin{aligned} g &= \mathbb{E}\big|\mathbb{E}[Y|\Psi(x), e_1] - \mathbb{E}[Y|\Psi(x), e_2]\big| = \mathbb{E}\big|\mathbb{E}[Y|a, e_1] - \mathbb{E}[Y|a, e_2]\big| \\ &= \frac{1}{2}\big|\mathbb{E}[Y|a=0, e_1] - \mathbb{E}[Y|a=0, e_2]\big| + \frac{1}{2}\big|\mathbb{E}[Y|a=1, e_1] - \mathbb{E}[Y|a=1, e_2]\big| \\ &= \frac{1}{2}\big(\big|0.9993(0) - 0.0495(0)\big| + \big|0.0031(1) - 0.9864(1)\big|\big) + \frac{1}{2}\big(\big|0.0007(0) - 0.9505(0)\big| + \big|0.9969(1) - 0.0136(1)\big|\big) = 0.98 \end{aligned}$$

EIIL for SquareMNIST: According to Table A1, the group sufficiency gap can be computed as follows:

$$\begin{aligned} g &= \mathbb{E}\big|\mathbb{E}[Y|\Psi(x), e_1] - \mathbb{E}[Y|\Psi(x), e_2]\big| = \mathbb{E}\big|\mathbb{E}[Y|a, e_1] - \mathbb{E}[Y|a, e_2]\big| \\ &= \frac{1}{2}\big|\mathbb{E}[Y|a=0, e_1] - \mathbb{E}[Y|a=0, e_2]\big| + \frac{1}{2}\big|\mathbb{E}[Y|a=1, e_1] - \mathbb{E}[Y|a=1, e_2]\big| \\ &= \frac{1}{2}\big(\big|0.8516(0) - 0.0968(0)\big| + \big|0.1515(1) - 0.8963(1)\big|\big) + \frac{1}{2}\big(\big|0.1484(0) - 0.9032(0)\big| + \big|0.8485(1) - 0.1037(1)\big|\big) = 0.74 \end{aligned}$$

References

1. Geirhos, R.; Jacobsen, J.H.; Michaelis, C.; Zemel, R.; Brendel, W.; Bethge, M.; Wichmann, F.A. Shortcut learning in deep neural networks. *Nat. Mach. Intell.* **2020**, *2*, 665–673. [CrossRef]
2. Mehrabi, N.; Morstatter, F.; Saxena, N.; Lerman, K.; Galstyan, A. A survey on bias and fairness in machine learning. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–35. [CrossRef]
3. Arjovsky, M.; Bottou, L.; Gulrajani, I.; Lopez-Paz, D. Invariant risk minimization. *arXiv* **2019**, arXiv:1907.02893.
4. Arpit, D.; Jastrzębski, S.; Ballas, N.; Krueger, D.; Bengio, E.; Kanwal, M.S.; Maharaj, T.; Fischer, A.; Courville, A.; Bengio, Y.; et al. A closer look at memorization in deep networks. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, NSW, Australia, 6–11 August 2017; pp. 233–242.
5. 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* **2019**, arXiv:1911.08731.

6. Creager, E.; Jacobsen, J.H.; Zemel, R. Environment inference for invariant learning. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 18–24 July 2021; pp. 2189–2200.
7. 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 Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 18–24 July 2021; pp. 6781–6792.
8. Howard, F.M.; Dolezal, J.; Kochanny, S.; Schulte, J.; Chen, H.; Heij, L.; Huo, D.; Nanda, R.; Olopade, O.I.; Kather, J.N.; et al. The impact of site-specific digital histology signatures on deep learning model accuracy and bias. *Nat. Commun.* **2021**, *12*, 1–13. [CrossRef] [PubMed]
9. Larrazabal, A.J.; Nieto, N.; Peterson, V.; Milone, D.H.; Ferrante, E. Gender imbalance in medical imaging datasets produces biased classifiers for computer-aided diagnosis. *Proc. Natl. Acad. Sci. USA* **2020**, *117*, 12592–12594. [CrossRef] [PubMed]
10. Oakden-Rayner, L.; Dunnmon, J.; Carneiro, G.; Ré, C. Hidden stratification causes clinically meaningful failures in machine learning for medical imaging. In Proceedings of the ACM Conference on Health, Inference, and Learning, Toronto, ON, Canada, 2–4 April 2020; pp. 151–159.
11. Buolamwini, J.; Gebru, T. Gender shades: Intersectional accuracy disparities in commercial gender classification. In Proceedings of the Conference on Fairness, Accountability and Transparency, PMLR, New York, NY, USA, 23–24 February 2018; pp. 77–91.
12. Krco, N.; Laugel, T.; Loubes, J.M.; Detyniecki, M. When Mitigating Bias is Unfair: A Comprehensive Study on the Impact of Bias Mitigation Algorithms. *arXiv* **2023**, arXiv:2302.07185.
13. Nam, J.; Cha, H.; Ahn, S.; Lee, J.; Shin, J. Learning from failure: De-biasing classifier from biased classifier. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 20673–20684.
14. Krasanakis, E.; Spyromitros-Xioufis, E.; Papadopoulos, S.; Kompatsiaris, Y. Adaptive sensitive reweighting to mitigate bias in fairness-aware classification. In Proceedings of the 2018 World Wide Web Conference, Lyon, France, 23–27 April 2018; pp. 853–862.
15. Tzeng, E.; Hoffman, J.; Saenko, K.; Darrell, T. Adversarial discriminative domain adaptation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7167–7176.
16. Long, M.; Cao, Z.; Wang, J.; Jordan, M.I. Conditional adversarial domain adaptation. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 1647-–1657.
17. 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 Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 18–24 July 2021; pp. 5815–5826.
18. Srivastava, M.; Hashimoto, T.; Liang, P. Robustness to spurious correlations via human annotations. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 13–18 July 2020; pp. 9109–9119.
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* **2021**, arXiv:2102.06406.
20. Rosenfeld, E.; Ravikumar, P.; Risteski, A. Domain-adjusted regression or: Erm may already learn features sufficient for out-of-distribution generalization. *arXiv* **2022**, arXiv:2202.06856.
21. Idrissi, B.Y.; Arjovsky, M.; Pezeshki, M.; Lopez-Paz, D. Simple data balancing achieves competitive worst-group-accuracy. In Proceedings of the Conference on Causal Learning and Reasoning, PMLR, Eureka, CA, USA, 11–13 April 2022; pp. 336–351.
22. Kirichenko, P.; Izmailov, P.; Wilson, A.G. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv* **2022**, arXiv:2204.02937.
23. Bao, Y.; Chang, S.; Barzilay, R. Predict then interpolate: A simple algorithm to learn stable classifiers. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 18–24 July 2021; pp. 640–650.
24. Sohoni, N.; Dunnmon, J.; Angus, G.; Gu, A.; Ré, C. No subclass left behind: Fine-grained robustness in coarse-grained classification problems. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 19339–19352.
25. Zhang, J.; Lopez-Paz, D.; Bottou, L. Rich feature construction for the optimization-generalization dilemma. In Proceedings of the International Conference on Machine Learning, PMLR, Baltimore, MD, USA, 17–23 July 2022; pp. 26397–26411.
26. Lahoti, P.; Beutel, A.; Chen, J.; Lee, K.; Prost, F.; Thain, N.; Wang, X.; Chi, E. Fairness without demographics through adversarially reweighted learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 728–740.
27. Yong, L.; Zhu, S.; Tan, L.; Cui, P. ZIN: When and How to Learn Invariance Without Environment Partition? *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 24529–24542.
28. Liu, J.; Hu, Z.; Cui, P.; Li, B.; Shen, Z. Heterogeneous risk minimization. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 18–24 July 2021; pp. 6804–6814.
29. Matsuura, T.; Harada, T. Domain generalization using a mixture of multiple latent domains. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 11749–11756.
30. Bae, J.H.; Choi, I.; Lee, M. Meta-learned invariant risk minimization. *arXiv* **2021**, arXiv:2103.12947.
31. Lin, Y.; Dong, H.; Wang, H.; Zhang, T. Bayesian invariant risk minimization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 16021–16030.
32. Wald, Y.; Feder, A.; Greenfeld, D.; Shalit, U. On calibration and out-of-domain generalization. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 2215–2227.
33. Lin, Y.; Zhu, S.; Cui, P. ZIN: When and How to Learn Invariance by Environment Inference? *arXiv* **2022**, arXiv:2203.05818.

34. Yao, H.; Wang, Y.; Li, S.; Zhang, L.; Liang, W.; Zou, J.; Finn, C. Improving out-of-distribution robustness via selective augmentation. In Proceedings of the International Conference on Machine Learning, PMLR, Baltimore, MD, USA, 17–23 July 2022; pp. 25407–25437.
35. Shi, Y.; Seely, J.; Torr, P.H.; Siddharth, N.; Hannun, A.; Usunier, N.; Synnaeve, G. Gradient matching for domain generalization. *arXiv* **2021**, arXiv:2104.09937.
36. Koyama, M.; Yamaguchi, S. Out-of-distribution generalization with maximal invariant predictor. *arXiv* **2020**, arXiv:2008.01883.
37. Rame, A.; Dancette, C.; Cord, M. Fishr: Invariant gradient variances for out-of-distribution generalization. In Proceedings of the International Conference on Machine Learning, PMLR, Baltimore, MD, USA, 17–23 July 2022; pp. 18347–18377.
38. Sagawa, S.; Raghunathan, A.; Koh, P.W.; Liang, P. An investigation of why overparameterization exacerbates spurious correlations. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 13–18 July 2020; pp. 8346–8356.
39. Zhang, J.; Menon, A.; Veit, A.; Bhojanapalli, S.; Kumar, S.; Sra, S. Coping with label shift via distributionally robust optimisation. *arXiv* **2020**, arXiv:2010.12230.
40. Ben-Tal, A.; El Ghaoui, L.; Nemirovski, A. *Robust Optimization*; Princeton University Press: Princeton, NJ, USA, 2009; Volume 28.
41. Ganin, Y.; Lempitsky, V. Unsupervised domain adaptation by backpropagation. In Proceedings of the International Conference on Machine Learning, PMLR, Lille, France, 7–9 July 2015; pp. 1180–1189.
42. Zhao, H.; Des Combes, R.T.; Zhang, K.; Gordon, G. On learning invariant representations for domain adaptation. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 9–15 June 2019, pp. 7523–7532.
43. Pezeshki, M.; Kaba, O.; Bengio, Y.; Courville, A.C.; Precup, D.; Lajoie, G. Gradient starvation: A learning proclivity in neural networks. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 1256–1272.
44. Zhang, Z.; Sabuncu, M. Generalized cross entropy loss for training deep neural networks with noisy labels. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 8792–8802.
45. Rockafellar, R.T.; Uryasev, S. Optimization of Conditional Value-at-Risk. *J. Risk* **2000**, *2*, 21–42. [CrossRef]
46. Sun, B.; Saenko, K. Deep coral: Correlation alignment for deep domain adaptation. In Proceedings of the Computer Vision–ECCV 2016 Workshops, Amsterdam, The Netherlands, 8–10 and 15–16 October 2016; Proceedings, Part III 14, pp. 443–450.