LEARNING AN INVERTIBLE OUTPUT MAPPING CAN MITIGATE SIMPLICITY BIAS IN NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep Neural Networks are known to be brittle to even minor distribution shifts compared to the training distribution. While one line of work has demonstrated that *Simplicity Bias* (SB) of DNNs – bias towards learning only the simplest features – is a key reason for this brittleness, another recent line of work has surprisingly found that diverse/ complex features are indeed learned by the backbone, and their brittleness is due to the linear classification head relying primarily on the simplest features. To bridge the gap between these two lines of work, we first hypothesize and verify that while SB may not altogether preclude learning complex features, it amplifies simpler features over complex ones. Namely, simple features are replicated several times in the learned representations while complex features might not be replicated. This phenomenon, we term *Feature Replication Hypothesis*, coupled with the *Implicit Bias* of SGD to converge to maximum margin solutions in the feature space, leads the models to rely mostly on the simple features for classification. To mitigate this bias, we propose *Feature Reconstruction Regularizer (FRR)* to ensure that the learned features can be reconstructed back from the logits. The use of *FRR* in linear layer training (*FRR-L*) encourages the use of more diverse features for classification. We further propose to finetune the full network by freezing the weights of the linear layer trained using *FRR-L*, to refine the learned features, making them more suitable for classification. Using this simple solution, we demonstrate up to 15% gains in OOD accuracy on the recently introduced semi-synthetic datasets with extreme distribution shifts. Moreover, we demonstrate noteworthy gains over existing SOTA methods on the standard OOD benchmark DomainBed as well.

1 INTRODUCTION

Despite the remarkable success of Deep Neural Networks (DNNs) in various fields, they are known to be brittle against even minor shifts in the data distribution during inference, which are not uncommon in a real world setting (Quinonero-Candela et al., 2008; Torralba & Efros, 2011). For example, a self-driving car that works well in normal weather may perform poorly when it is snowing, leading to disastrous outcomes. The need for improving the robustness of such systems against distribution shifts has sparked interest in the area of Out-Of-Distribution or OOD generalization (Hendrycks & Dietterich, 2019; Gulrajani & Lopez-Paz, 2020).

In this work, we aim to tackle the problem of OOD generalization of Neural Networks in a covariate-shift (Shimodaira, 2000) based classification setting, by addressing the fundamental cause of their brittleness, rather than by explicitly enforcing invariances in the network using domain labels or data augmentations. More specifically, we aim to mitigate the issue of *Simplicity Bias*, which is the tendency of *Stochastic Gradient Descent (SGD)* based solutions to overly rely on simple features alone, rather than on a diverse set of features (Arpit et al., 2017; Valle-Perez et al., 2018). While this behavior was earlier used to explain the remarkable generalization of Deep Networks, recent works suggest that this is indeed a key reason behind their brittleness to domain shifts (Shah et al., 2020).

The extent of Simplicity Bias seen in models is a result of two important factors - diversity of features learned by the feature extractor[1], and the extent to which these diverse features are used for the task

[1]In this paper, we refer to the penultimate layer's activations as *features*.

at hand, such as classification[2]. Recent works suggest that generalization to distribution shifts can be improved by retraining the last layer alone, indicating that the features learned may already be good enough for the same (Rosenfeld et al., 2022; Kirichenko et al., 2022b). Does this imply that brittleness of models can be attributed to the learning of the classification head alone? If this is the case, why does SGD fail to utilize these diverse features despite its *Implicit Bias* to converge to a maximum margin solution in a linearly separable case (Soudry et al., 2018)? To answer these questions, we firstly hypothesize and empirically verify that Simplicity Bias leads to the learning of simple features over and over again, as compared to other, more complex features. For example, among the 512 penultimate layer features of a ResNet, 462 of them might capture a simple feature such as *color*, while the remaining 50 might capture a more complex feature such as *shape* – we refer to this as *(Simple) Feature Replication Hypothesis*. Assuming feature replication hypothesis, we further show theoretically and empirically that a maximum margin classifier in the replicated feature space would give much higher importance to the replicated feature when compared to others, highlighting why the linear layer relies more on simpler features for classification.

To mitigate this, we propose a novel regularizer termed *Feature Reconstruction Regularizer (FRR)*, to enforce that the features learned by the network can be reconstructed back from the logit or pre-softmax layer used for the classification task. As shown in Fig.2, we firstly propose to train the linear classifier alone by freezing the weights of the feature extractor. This formulation enables the learning of an *Invertible Mapping* in the output layer, specifically for the domain of features seen during training. This further allows the logit layer to act as an information bottleneck, encouraging all the factors of variation in the features to be utilized for the classification task, thereby improving the diversity of features used. We theoretically show that adding this constraint while finetuning the linear layer can learn a max-margin classifier in the original input space, disregarding feature replication. Consequently, the learnt linear classifier also gives more importance to non replicated complex features while making predictions. We further explore the possibility of improving the quality of features learned by the feature extractor, by using *FRR* for finetuning the backbone as well. In order to do this, we freeze the linear classification head, and further finetune the backbone with FRR. We find that this encourages the network to indeed learn better quality features that are more relevant for classification. We list the key contributions of this work below -

- *Key Observation*: We provide a crisp hypothesis of "feature replication" to explain the brittleness of ERM trained neural networks to OOD data (Sec 3.1). Using this, we further provide theoretical and empirical evidence to justify the existence of Simplicity Bias in maximum margin classifiers.
- *Novel Algorithm based on the Observation*: Based on this, we introduce a novel FRR regularizer to safeguard against the feature replication phenomenon (Sec 3.2). We also provide theoretical support for FRR in an intuitive data distribution setting. Furthermore, we introduce a simple *FRR-L* method to only regularize the linear head with FRR, and then introduce *FRR-FLFT* training regimen to train the feature extractor for improved OOD robustness (Sec 4).
- *Empirical validation of the hypothesis and the proposed algorithm*: We demonstrate the effectiveness of FRR-FLFT and FRR-L by conducting extensive experiments on semi-real datasets (Table 2) constructed to study OOD brittleness, as well as on standard OOD generalization benchmarks, where FRR-FLFT can provide up to 3% gains over SOTA methods for OOD generalization(Table 3).

2 Related Works

Learning diverse classifiers to counter simplicity bias: Recent works have shown that ERM trained models learn diverse features, however, the linear layer fails at capturing and utilizing these diverse features properly. There have been several attempts at training classifiers which can make use of such diverse features. Teney et al. (2022) train a number of linear classifiers on top of a pre-trained network with a diversity regularizer, which encourages the classifiers to rely on different features. Xu et al. (2022) extend this idea further to train a classifier which is "orthogonal" to the original network, i.e. its predictions do not depend on the same features that the original network's predictions depend on. However, this assumes access to an unbiased network, whose predictions already take both sets of features into account. Kirichenko et al. (2022a) show that reweighting train set examples and retraining the last layer of a pre-trained deep network can alleviate spurious cor-

[2]i.e., by the final classification layer.

Table 1: **Features replication in Coloured MNIST**: We observe that ERM learns more colour features than shape features, and the prediction is less correlated with the shape features. Adding FRR makes the network depend more on shape and less on colour, leading to better OOD performance.

	Number		Correlation with output			
Algorithm	Colour	Shape	Colour	Shape	ID Accuracy	OOD Accuracy
ERM	26	4	0.81	0.61	99.9%	59.1%
ERM+FRR-L	26	4	0.71	0.65	99.6%	64.9%

relations, provided one can access a balanced dataset. In contrast to these methods, our method can work simply on the training set data, and produce a single classifier which is debiased. Huang et al. (2020) propose to mute the features with highest gradients, and use only the other features to make a prediction. While this method suppresses the maximally used features, it does not encourage the learning of hard-to-learn features, which is directly realized using our loss formulation. Kumar et al. (2022) suggest that finetuning the final linear layer first before finetuning the entire network can make it more robust to OOD shifts, and we utilize this insight in the FRR-FLFT phase of our method. A complementary approach to this problem is to learn features that are more diverse (Zhang et al., 2022). We note that applying our proposed method on top of such techniques would encourage the classifier to use the diverse features effectively, and this can further benefit the performance.

Domain Generalization and OOD robustness: The performance of neural networks is known to drop when there is a mismatch in the train and test distributions (Hendrycks & Dietterich, 2019), and methods to mitigate this have been gaining a lot of attention in recent years. The problem has been studied under various assumptions on distribution shift. The commonly studied setting of domain generalization (Gulrajani & Lopez-Paz, 2020; Li et al., 2018a) assumes that the train distribution consists of a mixture of distinct distributions (called domains), with each train sample having a domain label associated with it. The stronger setting of aggregate domain generalization (Thomas et al., 2021; Matsuura & Harada, 2020) assumes training data to be drawn from a mixture of distributions, but does not assume the availability of domain labels. Finally, OOD robustness (Hendrycks & Dietterich, 2019; Koh et al., 2021) drops all of these assumptions. Most works tackling the domain generalization problem attempt to train a model whose predictions are invariant to the domain label (Li et al., 2018a; Arjovsky et al., 2019), or try to align the features of the model for examples from different domains (Shi et al., 2021; Shankar et al., 2018). However, since we aim to tackle the stronger setting of OOD generalization, we do not use domain labels. Tackling the OOD robustness problem, Thomas et al. (2021) and Matsuura & Harada (2020) first cluster training examples into "pseudo-domains", after which standard domain generalization techniques are used. Another recent line of works propose using model averaging (Cha et al., 2021; Li et al., 2022) and/or ensembling (Arpit et al., 2021) for better OOD generalization. These techniques are complementary to our contribution, and we demonstrate how they can benefit each other in our empirical evaluation.

3 Feature Replication Hypothesis

Prior works have shown that neural networks trained with SGD exhibit simplicity bias (SB), even when initialized with pre-trained models that can capture complex features. Our Feature Replication Hypothesis – FRH– states that: SB is observed because the simpler features of the input are *replicated* multiple times in the feature space of neural networks. When trained using SGD, the final linear layer then learns the max margin classifier on these replicated features, which leads to over-reliance on simpler features in the input. Hence, the outputs of the network are brittle to distribution shifts that change such replicated features. In this section, we provide empirical and theoretical evidence for FRH, and propose a new regularizer – FRR– to mitigate this effect.

We first introduce some useful notations. Let $f_\theta(x) : \mathbb{R}^d \to \mathbb{R}^m$ be the feature extractor of a neural network parameterized by weights θ, and $W \in \mathbb{R}^{m\times k}$ be the weight matrix of the linear classifier. For input $x \in \mathbb{R}^d$, the output of the network is $W^T f_\theta(x) \in \mathbb{R}^k$.

3.1 EMPIRICAL VALIDATION OF FEATURE REPLICATION HYPOTHESIS (FRH) IN ERM

Coloured MNIST dataset. To empirically demonstrate feature replication, we use a binarized version of the coloured MNIST dataset (Gulrajani & Lopez-Paz, 2020). We construct this dataset by first assigning two digits of the MNIST dataset, namely "1" and "5", to classes 0 and 1 respectively. While training the network, we super-impose images of "1" onto colours of range $R_0 = [(115, 0, 0) - (256, 141, 0)]$ (i.e. red), and images of "5" onto colours of range $R_1 = [(0, 115, 0) - (141, 256, 0)]$ (i.e. green). The dataset is constructed such that the simple feature, namely colour, is weakly correlated with the labels, while the complex shape features are strongly correlated with labels. See Appendix B.1 for more details about the dataset.

Training setup: We train a model on this dataset, and test it on images which do not have any correlation between the label and the colour, i.e. images where the digits "1" and "5" are superimposed on randomly coloured backgrounds. We construct this test distribution to see how well different algorithms learn simple (i.e. colour) and complex (i.e. shape) features, since an algorithm which depends only on the spurious colour features would not have good performance on the test domain. We train a four layered CNN on this data. If a feature in the penultimate layer $f_\theta(x)$ has more than 90% correlation with the color or shape of the input, then we call it as a color feature or a shape feature, respectively. We also compute the correlation of these features with the output of the network ($W^T f_\theta(x)$) over inputs from the test domain. This gives us information of the learnt features, and their contributions to the final prediction of the network. Note that the feature dimension is $m = 32$, and the output dimension is $k = 1$.

Observations: In Table 1, we report the number of colour features, shape features, and the average correlation of each of these with the final prediction. We observe that the ERM trained model learns both shape and colour features, but the number of learnt colour features (26) is much higher than the number of shape features (4), despite their weaker correlation with labels, thus validating our *Feature Replication Hypothesis*. We also visualize the inter-feature correlation of the learnt features in Fig 4, which shows blocks of highly correlated features, further validating our hypothesis. Furthermore, we note that correlation of the output with the shape features is lower, leading to OOD accuracy of 59% only.

3.2 FEATURE RECONSTRUCTION REGULARIZER (FRR)

To alleviate simple feature replication issue, we propose *Feature Reconstruction Regularizer* (FRR) to enforce that the learned features can be reconstructed from the output logits. We propose to retrain the final linear layer using this regularizer to allow the model to utilize diverse features to compute the final output. We implement this by introducing another neural network with the objective of reconstructing the features of the network from the output logits, i.e. features $f_\theta(x)$ should be recoverable from the predictions of the network through a transform $\mathcal{T}_\phi(.)$ parameterized by ϕ. That is, FRR is given by:

$$\mathcal{L}_{\text{FRR}}(x, \theta, W, \phi) = ||f_\theta(x) - \mathcal{T}_\phi(W^T f_\theta(x))||_p \tag{1}$$

where $||.||_p$ denotes the ℓ_p norm. We set this norm to be ℓ_∞ or ℓ_1 in our experiments. In the simplest case, $\mathcal{T}_\phi(y) = \phi y$, where $\phi \in \mathbb{R}^{m \times k}$. Note that in order to find the appropriate ϕ, we jointly optimize W and ϕ using gradient descent based optimizers. We also experiment with ϕ being a more complex neural network.

We **empirically** validate FRR on **Coloured MNIST**, where using FRR with the linear layer leads to lower correlation with Colour compared to standard ERM (Table 2). Consequently, OOD accuracy improves by 5% over ERM.

3.3 FRH & FRR: THEORETICAL ANALYSIS

We now present a simple and intuitive data distribution with feature replication that highlights the OOD brittleness of standard ERM, and also demonstrates FRR can be significantly more robust.

Data Distribution: Consider a linearly separable distribution consisting of two factors of variation as shown in Figure 1. That is, consider the following distribution $(x, y) \sim \mathcal{D}$, where,

$$y = \pm 1 \text{ with probability } 0.5, \quad x = [y,\ y] + [n_1,\ n_2] \in \mathbb{R}^2, \quad n_i \sim \text{Unif}[-0.5, 0.5], i \in [2]. \tag{2}$$

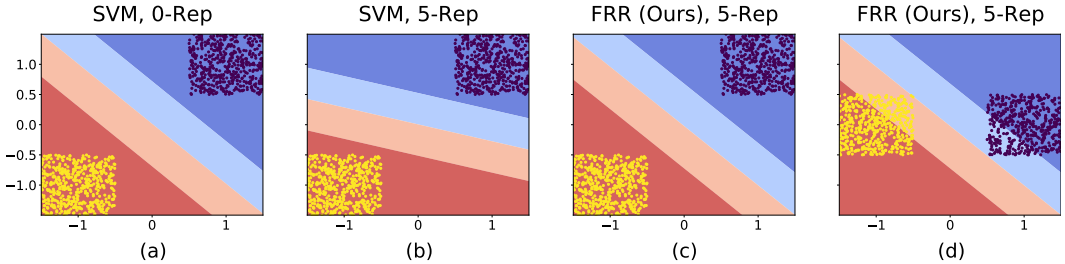

Figure 1: We demonstrate the brittleness of SVM (a, b) and effectiveness of FRR (c, d) based classifiers on a toy dataset comprising of 2 factors of variation, sampled from a uniform distribution. We consider d (= 0 or 5) feature replications (Rep) along the y-axis. FRR converges to a maximum margin solution in the non-replicated feature space, resulting in improved OOD robustness (d)

Also consider a feature extractor $f_\theta(.)$ which captures feature replication in the first feature, i.e. for every data point (x, y), the new, feature replicated data point will be $(\tilde{x}, y)$, where,

$$f_\theta(x) = \tilde{x} = [x_1, \cdots, x_1, x_2] \in \mathbb{R}^{d+1}, \tag{3}$$

i.e., x_1 is repeated d times. The joint distribution of features and labels is denoted by $\tilde{\mathcal{D}}$. Finally, we define the l_2 max margin classifier over a distribution $\mathcal{D}$ as $w_{\text{MM}} := \arg\min_w \frac{1}{2} \|w\|_2^2$ subject to $y \cdot \langle w, x \rangle \geq 1 \; \forall \; (x, y) \in \text{Supp}(\mathcal{D})$. Then we have the following results:

Claim 3.1 (Brittleness due to Feature Replication)**.** *Consider the data distribution given in Equation 2, 3. Then, the following holds: (1.) The max-margin classifier w_{MM} over $\mathcal{D}$ is given by $w_{MM} = [1, 1]$, and (2.) The max-margin classifier $\tilde{w}_{MM}$ over $\tilde{\mathcal{D}}$ is given by $\tilde{w}_{MM} = \left[\frac{2}{d+1}, \cdots, \frac{2}{d+1}\right] \in \mathbb{R}^{d+1}$.*

The above claim implies that when there are replicated features to the input of the linear layer, the max-margin classifier would give much more importance to the feature that is replicated. Hence, even a minor change in this replicated feature in the input space would be *amplified* in the output of the classifier. This is especially concerning in light of the observations in Table 1, which validate the Feature Replication Hypothesis in Coloured MNIST.

Proposition 3.2 (Robustness of FRR)**.** *Denote the average feature reconstruction loss $\mathcal{L}_{FRR}(\tilde{w}, \tilde{\phi}) := \mathbb{E}_{(\tilde{x}, y) \sim \tilde{\mathcal{D}}} \left[\left\| \langle \tilde{w}, \tilde{x} \rangle \tilde{\phi} - \tilde{x} \right\|_\infty^2 \right]$ and consider any $(\tilde{w}^*, \tilde{\phi}^*)$ satisfying:*

$$(\tilde{w}^*, \tilde{\phi}^*) \in \arg\min_{(\tilde{w}, \tilde{\phi})} \mathcal{L}_{FRR}(\tilde{w}, \tilde{\phi}) \text{ subject to } y \cdot \langle \tilde{w}, \tilde{x} \rangle \geq 0 \; \forall \; (\tilde{x}, y) \in \text{Supp}\left(\tilde{\mathcal{D}}\right).$$

We have that: $\tilde{w}_1^* + \cdots + \tilde{w}_d^* = \tilde{w}_{d+1}^*$. Consequently, we have $\langle \tilde{w}^*, \tilde{x} \rangle \propto \langle w_{MM}, x \rangle$ for all $x \in \mathbb{R}^2$.

Above result shows that the feature reconstruction regularizer will produce a linear classifier that gives equal weights to the replicated and non-replicated features. This is equivalent to a maximum margin classifier in the *non-replicated feature space*, thereby resulting in enhanced robustness to distribution shifts. Same is reflected in Figure 1 (c), (d) which show impact of FRR on the trained boundary in the non-replicated feature space. We defer the proofs of the above to Appendix A.

4 Training Procedure

Pretraining : In order to learn features which are relevant to the train distribution, we first pretrain our model using standard ERM with the cross-entropy loss $\mathcal{L}_{std}(W, \theta, (x, y))$.

FRR-L : Since ERM training is known to learn several rich and diverse features, we freeze the backbone parameters θ, and retrain the final layer W as the following-

$$(W_{\text{FRR}}, \phi_{\text{FRR}}) = \min_{W, \phi} \mathcal{L}_{\text{std}}(W, \theta, (x, y)) + \lambda_{\text{L}} \mathcal{L}_{\text{FRR}}(x, \theta, W, \phi) \tag{4}$$

where λ_L is a hyperparameter weighing the two losses. We train W and ϕ jointly. We refer to this step as FRR-L, i.e. Feature Reconstruction Regularizer - Linear, since we only train the linear layer.

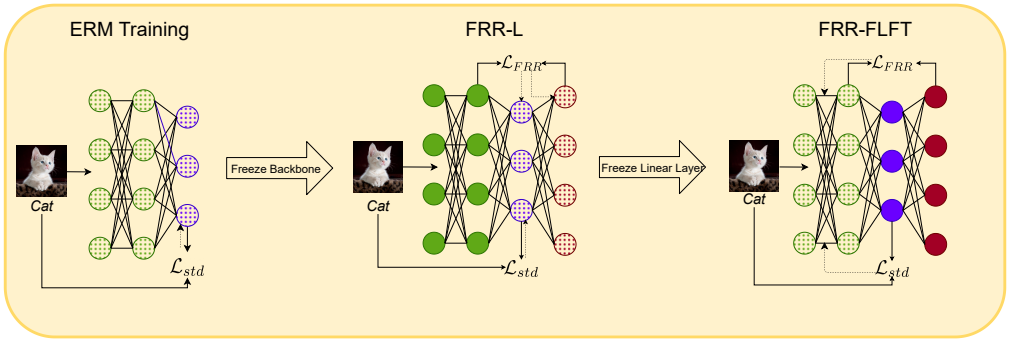

Figure 2: **Our training procedure**: Dotted fill indicates that the parameters are trainable.

FRR-FLFT : Following the suggestions of Kumar et al. (2022), we follow up the linear layer training with the finetuning of the feature extractor θ with a weighted combination of the cross-entropy loss and FRR, weighted by a hyper-parameter λ_{FLFT}. In this step we freeze the weights of the linear layer to improve the stability of training. We do this since naively using this constraint during network training could amplify the Simplicity Bias in networks in degenerate cases. For example, the backbone could learn to output a single replicated simple feature which is predictive enough on the training data. Reconstructing such a feature from logits would also be easy, but such a network might not generalize well. Formally, the optimization problem for this step is -

$$\theta_{\text{FLFT}} = \min_{\theta} \; \mathcal{L}_{\text{std}}(W_{\text{FRR}}, \theta, (x, y)) + \lambda_{\text{FLFT}} \mathcal{L}_{\text{FRR}}(x, \theta, W_{\text{FRR}}, \phi_{\text{FRR}}) \tag{5}$$

We view this step as "sharpening" the features for more accurate predictions. Freezing the linear head makes sure that the features do not collapse to a degenerate solution. Our training algorithm is summarized in Algorithm-1 and the training pipeline is illustrated in Figure 2.

5 Experimental Results

5.1 Understanding how FFR mitigates Simplicity Bias

To empirically illustrate the extent of Simplicity Bias in Neural Networks, Shah et al. (2020) introduced several synthetic and semi-synthetic datasets, where some features are explicitly *simple*, requiring a simpler decision boundary for prediction; while others are *complex*. In this section, we demonstrate the effectiveness of the proposed *Feature Reconstruction Regularizer* towards mitigating Simplicity Bias, by evaluating the same on a 10-class variant of the proposed semi-synthetic MNIST-CIFAR dataset, as discussed in the following section.

5.1.1 MNIST-CIFAR-10 dataset

We extend the simple binary MNIST-CIFAR dataset proposed by Shah et al. (2020) to a 10-class dataset, in order to evaluate the impact of the proposed *Feature Reconstruction Regularizer* in a more complex scenario when compared to the binary Colored-MNIST dataset presented in Section-3. We refer to this dataset as MNIST-CIFAR-10. The higher complexity of this dataset allows for a more reliable evaluation of various settings such as linear probing, full network finetuning and fixed-linear finetuning, with better granularity of results.

To construct this dataset, we first define correspondences between the classes of CIFAR-10 and MNIST. Each image from class *k* of MNIST is mapped with an image from class *k* of CIFAR-10, with the label being set to *k*. Thus, every training data sample (x_1, x_2, y) consists of x_1 and x_2, which are images from CIFAR-10 and MNIST respectively, along with their ground truth class y. It is to be noted that for both CIFAR-10 and MNIST, labels are always correlated with the respective images. In such a scenario, although a classifier can achieve very good performance by relying solely on the simple (MNIST) features, the goal of Out-Of-Distribution (OOD) robustness requires it to rely on the complex (CIFAR-10) features as well. This dataset represents the toughest setting of OOD generalization, where there is no differentiation between important features and spurious

Table 2: ID and OOD accuracy (%) by training on MNIST-CIFAR-10 in various training regimes.

Initialization	Layers trained	Exp ID	Training Loss	Training Dataset	In-Distribution (ID) MNIST-CIFAR-10	Out-Of-Distribution (OOD) MNIST-AvgCIFAR	Out-Of-Distribution (OOD) CIFAR-AvgMNIST
Random	All layers	E1	M1: ERM (CE)	MNIST-CIFAR-10	99.84 $\pm$ 0.01	97.44 $\pm$ 0.89	51.92 $\pm$ 1.52
		E2	Cross-Entropy	CIFAR-RandMNIST	88.53 $\pm$ 0.15	9.77 $\pm$ 0.11	88.52 $\pm$ 0.14
		E3	Cross-Entropy	MNIST-RandCIFAR	99.68 $\pm$ 0.02	94.84 $\pm$ 1.18	10.02 $\pm$ 0.13
M1:ERM	Linear layer	E4	M2: ERM-L (CE)	MNIST-CIFAR-10	99.86 $\pm$ 0.01	97.06 $\pm$ 0.05	52.73 $\pm$ 0.08
		E5	Cross-Entropy	CIFAR-RandMNIST	65.14 $\pm$ 0.05	10.15 $\pm$ 0.01	65.10 $\pm$ 0.03
		E6	Cross-Entropy	MNIST-RandCIFAR	99.71 $\pm$ 0.00	94.84 $\pm$ 0.04	10.33 $\pm$ 0.17
		E7	CE + Full-Rank Reg	MNIST-CIFAR-10	99.86 $\pm$ 0.01	97.04 $\pm$ 0.15	52.87 $\pm$ 1.09
		E8	M3: FRR-L (**Ours**)	MNIST-CIFAR-10	99.88 $\pm$ 0.00	96.81 $\pm$ 0.38	59.13 $\pm$ 0.37
M2:ERM-L	All layers	E9	Cross-Entropy	MNIST-CIFAR-10	99.84 $\pm$ 0.02	97.67 $\pm$ 0.19	53.33 $\pm$ 0.28
	Feature extractors	E10	Cross-Entropy		99.84 $\pm$ 0.01	97.67 $\pm$ 0.18	53.67 $\pm$ 0.40
	All layers	E11	FRR-FT		99.84 $\pm$ 0.02	97.32 $\pm$ 0.45	54.12 $\pm$ 0.44
	Feature extractors	E12	FRR-FLFT		99.81 $\pm$ 0.04	98.44 $\pm$ 0.64	60.02 $\pm$ 0.69
M3:FRR-L	All layers	E13	Cross-Entropy	MNIST-CIFAR-10	99.87 $\pm$ 0.01	97.03 $\pm$ 0.35	61.75 $\pm$ 0.33
	Feature extractors	E14	Cross-Entropy		99.88 $\pm$ 0.02	97.35 $\pm$ 0.34	63.73 $\pm$ 0.62
	All layers	E15	FRR-FT		99.85 $\pm$ 0.01	99.30 $\pm$ 0.05	62.13 $\pm$ 0.42
	Feature extractors	E16	M4: FRR-FLFT (**Ours**)		99.84 $\pm$ 0.03	99.45 $\pm$ 0.03	68.12 $\pm$ 0.96
M4:FRR-FLFT	Linear layer	E17	Cross-Entropy	CIFAR-RandMNIST	79.92 $\pm$ 0.33	11.93 $\pm$ 0.09	77.35 $\pm$ 0.10
		E18	Cross-Entropy	MNIST-RandCIFAR	99.71 $\pm$ 0.00	99.46 $\pm$ 0.00	10.27 $\pm$ 0.12

correlations. A real-world example of such a case is the classification of swans versus bears, with the training dataset consisting of only white swans and black bears. Here the model could either rely on shape or color for classification. A classification network that relies solely on the simplest feature color, fails to generalize to the test set consisting of black swans and polar bears.

5.2 Training and Evaluation Settings

We consider two separate ResNet-18 (He et al., 2016) feature extractors for CIFAR-10 and MNIST respectively. The outputs of the Global Average Pooling (GAP) layers in each of the feature extractors are concatenated to form a 1024 dimensional vector, which is given as input to the linear classifier. This architecture allows the computation of accuracy based on either a combination of both CIFAR-10 and MNIST features, or based on features of only one of the datasets. For example, to evaluate the performance of the classifier based on CIFAR-10 features alone, we replace the 512 dimensional MNIST feature vector of each data sample with an average feature vector computed from all images in the MNIST dataset. We refer to this as the CIFAR-AvgMNIST dataset, while the corresponding one for MNIST is refered to as the MNIST-AvgCIFAR dataset. Similar to the work by Shah et al. (2020), we define two additional datasets, CIFAR-RandMNIST and MNIST-RandCIFAR, where images from one of the datasets (MNIST and CIFAR-10 respectively) are randomly shuffled with respect to their corresponding labels. The base training (E1, E2, E3) is done for 500 epochs, and the linear layer training / finetuning (E4 - E18) is done for 20 epochs, without any augmentations.

5.3 Experimental Results in various Training Regimes

We present the results of training on the MNIST-CIFAR-10 dataset using different algorithms in Table 2. The mean and standard deviation across five runs have been reported for each case.

ERM Training: By training a randomly initialized model on the MNIST-CIFAR-10 dataset using the cross-entropy loss (E1), we obtain an accuracy of 99.84% on its corresponding test split. While the accuracy of this model on the MNIST-avgCIFAR dataset is high (97.44%), its performance on the CIFAR-avgMNIST dataset is poor (51.92%), indicating that the model chooses to rely more on the simpler MNIST features, rather than a combination of both CIFAR and MNIST features.

While the performance on the CIFAR-avgMNIST and MNIST-avgCIFAR datasets is sufficient to understand the extent of CIFAR/ MNIST features used by the classification head, it does not give a clear picture on the features learned by the two feature extractors. To understand this, we reinitialize the linear classification head randomly, and train the same using CIFAR-RandMNIST (E5) and MNIST-RandCIFAR datasets (E6) respectively. We obtain an accuracy of 65.2% on the CIFAR-avgMNIST dataset in the former case, indicating that although the CIFAR features learned can possibly achieve 13% higher accuracy (w.r.t. E1), the bias in the classification head prevents them from participating in the classification task. The MNIST-avgCIFAR accuracy of the latter case is high as expected. An upper bound on CIFAR-10 and MNIST accuracy that can be achieved with the selected architecture and training strategy (without using any augmentations) can be seen in E2 (88.53%) and E3 (99.68%) respectively.

Training the Linear Classification Head: As discussed, while ERM training (E1) learns features that can be used for better OOD performance (E5), it does not effectively leverage these features for the classification task. We firstly explore the possibility of bridging the difference in the CIFAR-avgMNIST accuracy between E1 and E5 by merely retraining the linear layer. By reinitializing and naively retraining the linear layer with Cross-entropy loss, the accuracy on CIFAR-avgMNIST improves by less than 1% (E4). Using the proposed *Feature Reconstruction Regularizer (FRR)* for training the linear layer alone, the CIFAR-avgMNIST accuracy improves by 7.21% as shown in E8, demonstrating the effectiveness of the proposed regularizer in mitigating Simplicity Bias. We penalize the ℓ_∞ norm of difference in original features and their reconstruction in addition to the minimization of cross-entropy loss. The reconstruction based regularizer enforces the network to utilize both CIFAR and MNIST features for classification. Since this regularizer resembles an orthonormality constraint on the linear classification head, we additionally check the effectiveness of explicitly enforcing a full-rank constraint on the linear layer by minimizing the following: $||WW^T - I||_F$ (E7). We find that this is not effective in improving the overall accuracy, possibly because it enforces a very stringent constraint on the final classification layer. Contrary to this, the proposed *Feature Reconstruction Regularizer* allows more flexibility by imposing this constraint only on the domain of features seen during training. This accounts for the simple feature replication as well, enabling to view the logit layer as an information bottleneck in the reconstruction.

Finetuning (FT) and Fixed Linear Finetuning (FLFT): We explore the finetuning of a given base model in two settings - firstly by finetuning all layers in the network (denoted as FT or FineTuning), and secondly, by freezing the parameters of the linear classification head and finetuning only the feature extractors, which we refer to as FLFT or Fixed Linear FineTuning. By finetuning an ERM trained base model using either of the two strategies (E9 and E10), we observe gains of less than 1%. We observe similar gains even by finetuning the full network with FRR (E11). Contrary to this, by using FRR-FLFT even on the ERM trained network (E12), we obtain 7.29% improvement over the base model. This shows that, by allowing the full network to change while imposing the FRR constraint, the network can continue to rely on simple features, possibly by reducing the number of complex features learned by the feature extractor. However, by freezing the weights of the linear layer and further imposing this constraint, the network is forced to refine the CIFAR features that are already being used for prediction.

Combining FRR-L and FRR-FLFT: While we obtain similar order of gains ($\sim 7\%$) using both FRR-L and FRR-FLFT individually, the former improves the diversity of features being considered by the classification head, while the latter improves the quality of the features themselves. We therefore propose a training strategy that combines both FRR-L and FRR-FLFT. Using this, we obtain gains of 16.2% over the ERM baseline as shown in E16, indicating that the combination of FRR-L and FRR-FLFT has a compounding effect by firstly selecting diverse features, and further refining these features to be more useful for classification. Although FRR-L followed by FRR-FT (E15) is also effective, it has about 6% lesser gains when compared to the proposed approach of FRR-L + FRR-FLFT. We note that following up FRR-L with ERM-FT (E13) or ERM-FLFT (E14) also refines the learned features, making them more suitable for the classification task, yielding 2.6% and 4.6% gains respectively over FRR-L.

We verify the quality of features learned by the feature extractors after the proposed training strategy FRR-L + FRR-FLFT by reinitializing and retraining the linear classifier on CIFAR-RandMNIST (E17) and MNIST-RandCIFAR (E18) datasets respectively. We observe considerable gains of around 15% on MNIST-CIFAR-10 accuracy using CIFAR-RandMNIST training when compared to ERM (E5), demonstrating that the proposed approach not only results in more CIFAR features being used for classification, but also leads to the learning of better CIFAR features.

5.4 OOD Generalization in a Real World Setting

We show the efficacy of FRR towards improving OOD generalization on the DomainBed (Gulrajani & Lopez-Paz, 2020) benchmark. We use the performance of the model on in-domain validation data (i.e. the *in-domain* strategy by Gulrajani & Lopez-Paz (2020)) to select the best hyper-parameters, and report the average performance and standard deviation across 5 random seeds.

Baselines : We compare our method against standard ERM training, which has proven to be a frustratingly difficult baseline (Gulrajani & Lopez-Paz, 2020), and also against several state of the art methods on this benchmark - SWAD (Cha et al., 2021), MIRO (Cha et al., 2022) and SMA (Arpit

Table 3: Results on DomainBed: The bottom partition shows results of methods that perform model weight averaging. In both cases, with (top) and without (bottom) model weight averaging, the proposed approach outperforms existing methods.

Algorithm	PACS	VLCS	OfficeHome	TerraIncognita	DomainNet	**Avg.**
ERM	85.5 ± 0.1	77.5 ± 0.4	66.5 ± 0.2	46.1 ± 0.6	40.9 ± 0.1	63.3
IRM	83.5 ± 0.8	78.5 ± 0.5	64.3 ± 2.2	47.6 ± 0.8	33.9 ± 2.8	61.6
CORAL	86.2± 0.3	78.8± 0.6	68.7± 0.3	47.6± 1.0	41.5 ± 0.1	64.5
MIRO	85.4 ± 0.4	79.0 ± 0.0	70.5 ± 0.4	50.4 ± 1.1	44.3 ± 0.2	65.9
ERM+FRR-L	85.7 ± 0.1	76.6 ± 0.2	68.4 ± 0.2	53.7 ± 0.6	44.2 ± 0.1	65.7
ERM+FRR	87.5 ± 0.1	77.6 ± 0.3	69.4 ± 0.1	54.1 ± 0.6	45.1 ± 0.1	**66.8**
SMA	87.5 ± 0.2	78.2 ± 0.2	70.6 ± 0.1	50.3 ± 0.5	46.0 ± 0.1	66.5
SWAD	88.1 ± 0.1	79.1 ± 0.1	70.6 ± 0.2	50.0 ± 0.3	46.5 ± 0.1	66.9
SWAD+FRR	89.2 ± 0.4	80.0 ± 0.2	70.3 ± 0.1	53.2 ± 0.3	46.2 ± 0.0	**67.9**

et al., 2021). Finally, we show that our approach can be effectively integrated with stochastic weight averaging to obtain further gains. See Appendix E for further experimental details.

Main Results: The main results of our algorithm are reported in Table 3. We find that our pipeline of training and finetuning with FRR, when combined with ERM achieves improved performance with respect to the state of the art methods that do not use model weight-averaging, and in fact achieves comparable performance to methods that use model weight averaging as well. Further, our method obtains substantial gains of more than 3% over ERM across datasets. The gains are especially pronounced for the larger datasets including DomainNet and TerraIncognita (8% and 5% resp.), indicating the efficacy and scalability of our method. Further, it is clear from Table 3 that finetuning the feature extractor once the linear layer is fixed provides a boost of over 1% on average over FRR-L. This empirically validates our finetuning paradigm which we denote as ERM+FRR. Finally, using our method in tandem with SWAD helps us achieve a new state-of-the-art on the DomainBed benchmark, outperforming other methods on three datasets while achieving comparable performance on two, and being better than existing SOTA by close to 1% on average. We report detailed results and further ablations in Appendices F and G.

6 Conclusion and Discussion

In this work, we consider the problem of OOD generalization through the lens of mitigating Simplicity Bias in Neural Network training. To unravel the paradox pertaining to the existence of Simplicity Bias in learning only the simplest features, and the observation that the features learned by large practical models may already be sufficiently diverse, we put forth the *Feature Replication Hypothesis* that conjectures the learning of replicated simple features and spare complex ones. Combining this with the *Implicit Bias* of SGD to converge to maximum margin solutions, we provide a theoretical justification to the high OOD sensitivity of Neural Networks.

To specifically overcome the effect of simple feature replication in linear layer training, we propose the *Feature Reconstruction Regularizer*, that penalizes the l_p norm distance between the features and their reconstruction from the output logits, thus improving the diversity of features used for classification. We further propose to freeze the weights of the linear layer thus trained, and use the FRR regularizer for finetuning the full network, to refine the features to be more useful for the downstream task. We justify the proposed regularizer both theoretically and empirically on synthetic and semi-synthetic datasets, and demonstrate its effectiveness in a real world OOD generalization setting.

We believe and hope that this work can pave the way towards obtaining a better understanding on the underlying causes for OOD brittleness of neural networks, and inspire the development of better algorithms for addressing the same. We believe the proposed regularizer can potentially work effectively in several other settings that involve the use of linear layer training/finetuning, such as domain adaptation and transfer learning. While the regularizer works effectively in a scenario where the network is first trained using an algorithm such as ERM to learn features that are *relevant* to the task at hand, the robustness of the proposed algorithm in the presence of severely non-relevant features is yet to be explored.

REFERENCES

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International conference on machine learning*, pp. 233–242. PMLR, 2017.

Devansh Arpit, Huan Wang, Yingbo Zhou, and Caiming Xiong. Ensemble of averages: Improving model selection and boosting performance in domain generalization. *arXiv preprint arXiv:2110.10832*, 2021.

Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. *Advances in neural information processing systems*, 24, 2011.

Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. Swad: Domain generalization by seeking flat minima. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Junbum Cha, Kyungjae Lee, Sungrae Park, and Sanghyuk Chun. Domain generalization by mutual-information regularization with pre-trained models. *arXiv preprint arXiv:2203.10789*, 2022.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

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

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Zeyi Huang, Haohan Wang, Eric P Xing, and Dong Huang. Self-challenging improves cross-domain generalization. In *European Conference on Computer Vision*, pp. 124–140. Springer, 2020.

Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv preprint arXiv:2204.02937*, 2022a.

Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv preprint arXiv:2204.02937*, 2022b.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pp. 5637–5664. PMLR, 2021.

Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054*, 2022.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018a.

Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 624–639, 2018b.

Ziyue Li, Kan Ren, Xinyang Jiang, Bo Li, Haipeng Zhang, and Dongsheng Li. Domain generalization using pretrained models without fine-tuning. *arXiv preprint arXiv:2203.04600*, 2022.

Toshihiko Matsuura and Tatsuya Harada. Domain generalization using a mixture of multiple latent domains. In *AAAI*, 2020.

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, pp. 10–18. PMLR, 2013.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

Joaquin Quinonero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. Mit Press, 2008.

Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. Domain-adjusted regression or: Erm may already learn features sufficient for out-of-distribution generalization. *arXiv preprint arXiv:2202.06856*, 2022.

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

Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. *arXiv preprint arXiv:1804.10745*, 2018.

Yuge Shi, Jeffrey Seely, Philip HS Torr, N Siddharth, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. Gradient matching for domain generalization. *arXiv preprint arXiv:2104.09937*, 2021.

Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19 (1):2822–2878, 2018.

Damien Teney, Ehsan Abbasnejad, Simon Lucey, and Anton van den Hengel. Evading the simplicity bias: Training a diverse set of models discovers solutions with superior ood generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16761–16772, 2022.

Xavier Thomas, Dhruv Mahajan, Alex Pentland, and Abhimanyu Dubey. Adaptive methods for aggregated domain generalization, 2021.

Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In *CVPR 2011*, pp. 1521–1528, 2011. doi: 10.1109/CVPR.2011.5995347.

Guillermo Valle-Perez, Chico Q Camargo, and Ard A Louis. Deep learning generalizes because the parameter-function map is biased towards simple functions. *arXiv preprint arXiv:1805.08522*, 2018.

Yilun Xu, Hao He, Tianxiao Shen, and Tommi Jaakkola. Controlling directions orthogonal to a classifier. *arXiv preprint arXiv:2201.11259*, 2022.

Jianyu Zhang, David Lopez-Paz, and Léon Bottou. Rich feature construction for the optimization-generalization dilemma. *arXiv preprint arXiv:2203.15516*, 2022.

A PROOFS OF THEORETICAL RESULTS

Proof of claim 3.1 :

Proof. First, observe that $w = [1, 1]$ satisfies the constraint $y \cdot \langle w, x \rangle \geq 1 \ \forall \ (x, y) \in \text{Supp}\,(\mathcal{D})$. Now, consider any w satisfying this constraint. By considering $(y \cdot (0.5, 0.5), y) \in \text{Supp}\,(\mathcal{D})$, we have that $\frac{w_1+w_2}{2} \geq 1$. We now conclude by observing that among w which satisfy $\frac{w_1+w_2}{2} \geq 1$, $[1, 1]$ minimizes $\frac{1}{2}\|w\|_2^2$. The proof of the second part of the claim is identical to that of the first part. First, observe that $\tilde{w} = \left[\frac{2}{d+1}, \cdots, \frac{2}{d+1}\right]$ satisfies the constraint $y \cdot \langle \tilde{w}, \tilde{x} \rangle \geq 1 \ \forall \ (\tilde{x}, y) \in \text{Supp}\left(\tilde{\mathcal{D}}\right)$. Now, consider any $\tilde{w}$ satisfying this constraint. By considering $(y \cdot (0.5, 0.5, \cdots, 0.5), y) \in \text{Supp}\left(\tilde{\mathcal{D}}\right)$, we have that $\frac{\tilde{w}_1+\cdots+\tilde{w}_{d+1}}{2} \geq 1$. We now conclude by observing that among $\tilde{w}$ which satisfy $\frac{\tilde{w}_1+\cdots+\tilde{w}_{d+1}}{2} \geq 1$, $\left[\frac{2}{d+1}, \cdots, \frac{2}{d+1}\right]$ minimizes $\frac{1}{2}\|\tilde{w}\|_2^2$. □

The classifier $\tilde{w}_{\text{MM}} \in \mathbb{R}^{d+1}$ on $\tilde{x} \in \mathbb{R}^{d+1}$ is equivalent to the classifier $w_{\text{proj}} = \left[\frac{2d}{d+1}, \frac{2}{d+1}\right] \in \mathbb{R}^2$ on $x \in \mathbb{R}^2$:

$$\langle w_{proj}, x \rangle = \langle \tilde{w}_{\text{MM}}, \tilde{x} \rangle \ \forall \ x \in \mathbb{R}^2.$$

Proof of Proposition 3.2

Proof. Given any $\tilde{w}$, we first compute $\mathcal{L}_{\text{FRR}}(\tilde{w}) := \min_{\tilde{\phi}} \mathcal{L}_{\text{FRR}}(\tilde{w}, \tilde{\phi})$. To do so, we note that the minimizing $\tilde{\phi}$ for a given $\tilde{w}$, denoted by $\tilde{\phi}^*(\tilde{w})$ is given by:

$$\tilde{\phi}_i^*(\tilde{w}) = \arg\min_u \mathbb{E}_{(\tilde{x},y)\sim\tilde{\mathcal{D}}}[(\langle \tilde{w}, \tilde{x} \rangle u - \tilde{x}_i)^2].$$

So, $\tilde{\phi}_i^*(\tilde{w}) = \frac{\mathbb{E}[\langle \tilde{w}, \tilde{x} \rangle \tilde{x}_i]}{\mathbb{E}[\langle \tilde{w}, \tilde{x} \rangle^2]}$. So, we have that:

$$\min_u \mathbb{E}_{(\tilde{x},y)\sim\tilde{\mathcal{D}}}[(\langle \tilde{w}, \tilde{x} \rangle u - \tilde{x}_i)^2] = \mathbb{E}[\tilde{x}_i^2] - \frac{\mathbb{E}[\langle \tilde{w}, \tilde{x} \rangle \tilde{x}_i]^2}{\mathbb{E}[\langle \tilde{w}, \tilde{x} \rangle^2]}.$$

Since $\langle \tilde{w}, \tilde{x} \rangle = \tilde{w}_{1\to d} \cdot (y + n_1) + \tilde{w}_{d+1} \cdot (y + n_2)$, where $\tilde{w}_{1\to d} := \sum_{i=1}^d \tilde{w}_i$, we have:

$$\mathbb{E}[\langle \tilde{w}, \tilde{x} \rangle \tilde{x}_i] = \frac{13}{12} \cdot \begin{cases} \tilde{w}_{1\to d} & \text{if } 1 \leq i \leq d \\ \tilde{w}_{d+1} & \text{if } i = d+1 \end{cases},$$

where we used the fact that $\mathbb{E}[n_1^2] = \mathbb{E}[n_2^2] = 1/12$. Consequently, we have that $\mathcal{L}_{\text{FRR}}(\tilde{w}) = \frac{13}{12}\left(1 - \frac{\min(\tilde{w}_{1\to d}, \tilde{w}_{d+1})^2}{(\tilde{w}_{1\to d}+\tilde{w}_{d+1})^2 + \tilde{w}_{1\to d}^2/12 + \tilde{w}_{d+1}^2/12}\right)$. Consequently, any minimizer $\tilde{w}^*$ of $\mathcal{L}_{\text{FRR}}(\tilde{w})$ satisfies $\sum_{i=1}^d \tilde{w}_i = \tilde{w}_{1\to d} = \tilde{w}_{d+1}$. □

B SYNETHETIC DATASETS

B.1 COLOURED MNIST

In order to empirically demonstrate feature replication, we use a binarized version of the coloured MNIST dataset (Gulrajani & Lopez-Paz, 2020). To construct this dataset, we firstly assign two digits of the MNIST dataset, namely "1" and "5", to classes 0 and 1 respectively. For the in-domain training distribution, we associate colours in the range $R_0 = [(115, 0, 0) - (256, 141, 0))]$ (i.e. red) to label 0 (i.e. the digit "1") and the range $R_1 = [(0, 115, 0) - (141, 256, 0)]$ (i.e. green) to the label 1 (i.e. the digit "5"), where colors are represented in the RGB space. To summarize, while training the network, we super-impose images of "1" onto colours of range R_0, and images of "5" onto colours of range R_1. It is to be noted that the choice of colour ranges as defined above introduces an overlapping range between $[(115, 115, 0) - (141, 141, 0))]$ where images are associated with labels 0 and 1 with equal probability. This overlap reduces the correlation of colour features with labels, while shape features have a correlation of 1 with the labels. In Figure 3, we show examples of images from the train and test distributions of this dataset. In Figure 4, we pictorially depict the correlations between the 32 features learnt by the network. We can see a block structure emerging, indicating that there is a high amount of feature replication.

Figure 3: **Random images from the coloured MNIST dataset**: The top row shows examples from the train distribution, while the bottom row has images from the test distribution. Here, colour red corresponds to the digit 1 and green corresponds to the digit 5 in the train data, while this correlation is destroyed in the test data.

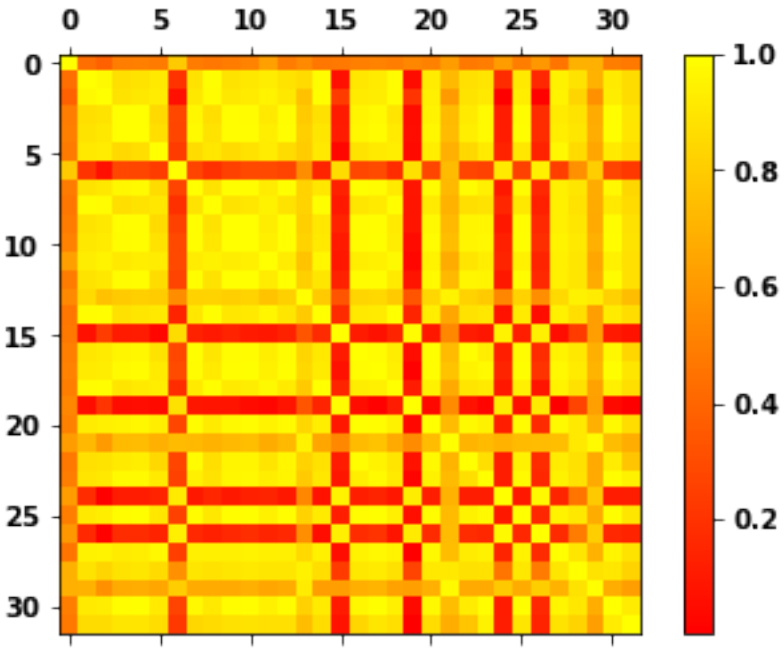

Figure 4: **Correlation of the features learnt on coloured MNIST**

B.2 Toy dataset

In line with the theoretical formulation described in Section-3.3, we further empirically validate the brittleness of SVM models and the highlight the effectiveness of the proposed *Feature Reconstruction Regularizer* in the presence of replicated features. We consider a linearly separable toy distribution consisting of two factors of variation as shown in Figure 1. We define the means of the two classes at (1,1) and (-1,-1) and construct 500 data points in each class by adding noise sampled from Unif[-0.5,0.5] independently along each dimension to the respective means. We sample an Out-of-Distribution (OOD) test set from a Uniform distribution with means centered at (1,0) and (-1,0) respectively, and similar noise along each dimension as the train set. Therefore, while the train distribution can be classified by considering either the features aligned with the X-coordinate or the Y-coordinate, the test set performance crucially depends on the variation along X-coordinate alone. We consider feature replication along the y-axis, and hence construct this OOD dataset to verify the extent to which the other feature is considered for classification. To select the best hyperparameter

for both SVM and FRR, we consider the presence of a validation set whose distribution is similar to the test distribution. As shown in Figure 1, we observe that the SVM model starts relying more on the replicated features alone in case of feature replication, compromising its performance on the OOD data. The proposed regularizer on the other hand, gives equal importance to both features even in the presence of feature replication, resulting in improved OOD generalization.

C ALGORITHM

Algorithm 1: Our training algorithm

Data: Training data $\mathcal{D}_S = \{(x_i, y_i) : i \in [n]\}$, model (θ, W), feature reconstruction model ϕ, $\lambda_{FRR}, \lambda_{FT}$

$\theta_{\text{std}}, W_{\text{std}} \leftarrow \text{Adam}\left(\min_{\theta,W} \sum_i \mathcal{L}_{\text{std}}(\theta, W, (x_i, y_i))\right)$.
`/* Standard training of model parameters` θ `and` W`. */`

Freeze θ to be θ_{std}
`/* Initializing model for training with FRR. */`

$W_{FRR}, \phi_{FRR} \leftarrow \text{Adam}\left(\min_{W,\phi} \sum_i \mathcal{L}_{\text{std}}(\theta_{std}, W, (x_i, y_i)) + \lambda_L \mathcal{L}_{FRR}(x_i, \theta_{std}, W, \phi)\right)$.
`/* FRR-L: Training` W, ϕ `with FRR defined in eqn. 4 */`

$\theta_{\text{FLFT}} \leftarrow \text{Adam}\left(\min_{\theta} \sum_i \mathcal{L}_{\text{std}}(\theta, W_{\text{FRR}}, (x_i, y_i)) + \lambda_{FLFT} \mathcal{L}_{FRR}(x_i, \theta, W_{FRR}, \phi_{FRR})\right)$.
`/* FRR-FLFT: Finetuning` θ `with FRR according to eqn. 5 */`

Result: Trained model $(\theta_{\text{FLFT}}, W_{\text{FRR}})$.

D DETAILS ON THE OOD GENERALIZATION SETTING CONSIDERED

The problem of improving robustness to distribution shifts has been studied in several settings, where, in addition to labeled source domain data, varying levels of access to the target domain data is assumed. Some of the well-researched settings include - *Unsupervised Domain Adaptation*, with access to only unlabeled target domain data (Pan & Yang, 2009; Ganin et al., 2016), and *Domain Generalization*, where typically data from several source distributions is assumed to be available, and the target domain in unseen during training (Blanchard et al., 2011; Li et al., 2018a; Gulrajani & Lopez-Paz, 2020)). In the latter case, it is assumed that all training data samples are annotated with domain labels as well, so that training algorithms can explicitly impose invariance to attributes that cause a distribution shift in input data without change in their label distribution (Muandet et al., 2013; Ganin et al., 2016; Li et al., 2018b; Arjovsky et al., 2019; Shi et al., 2021).

A more challenging case is when the training data belongs to several distributions that may not even be sufficiently discernable to have explicit domain annotations, or may contain multidimensional distribution shifts, such as weather, time of the day and geographical location, that cannot be easily annotated or clustered. We investigate this crucial setting which has been relatively less researched, and refer to it as *Aggregated Domain Generalization*, as introduced by Thomas et al. (2021). We note that this setting is different from the case of training on data from a single domain such as ImageNet, and evaluating on distribution shifts (Hendrycks & Dietterich, 2019), due to the availability of an *aggregate* of source domains during training, which can enable the effective use of in-domain validation set for hyperparameter selection.

While there have been several approaches to improve the performance of models in the setting of Domain Generalization, Gulrajani & Lopez-Paz (2020) show that when evaluated fairly, that is, without assuming access to the test domain data even for selecting the best set of hyperparameters, none of the approaches perform consistently better than standard training using *Empirical Risk Minimization (ERM)*. Furthermore, we consider the setting of *Aggregated Domain Generalization*, which is more challenging due to the absence of domain labels during both training and validation.

E EXPERIMENTAL DETAILS ON DOMAINBED

We test our approach on the DomainBed benchmark (Gulrajani & Lopez-Paz, 2020) comprising of five different datasets, each of which have k domains. For each dataset, we train a model on $k-1$ domains, and test it on the left out domain. The average out-of-domain performance across the k held-out domains is then reported. In this section we describe the hyper-parameter selection strategy

and the ranges considered for our approach. In line with the DomainBed testbench, we use ImageNet pretrained ResNet-50 models for all algorithms. We use random search to select hyperparameters for our algorithm, and use the suggested hyperparameters for the other baselines. We train for 3000 (5000 for DomainNet) steps in the FRR-L phase, and 5000 (10000 for DomainNet) steps in the FRR-FLFT phase. The batch size is fixed to 32, and SWAD hyper-parameters are the same as those used by Cha et al. (2021). We use the in-domain accuracy protocol from Gulrajani & Lopez-Paz (2020) to select hyper-parameters for each domain of each dataset, and search over 8 random combinations of hyper-parameters for each. The range of the hyperparameters is shown in Table 4. Note that we experiment with two implementations of ℓ_∞ norm: $\ell_{1,\infty}$, where we first compute the ℓ_∞ of feature reconstruction for each example in a batch and then average it across the batch, and $\ell_{\infty,1}$ where we compute the average ℓ_1 reconstruction norm of each feature across the batch, and then apply ℓ_∞ norm on this m dimensional vector. All our experiments were done on single V100 GPUs.

Table 4: **Ranges of hyperparameters considered for DomainBed**

Hparam	**Range**
Learning Rate	loguniform$(10^{-5}, 10^{-1})$
λ_{FRR}	loguniform$(10^{-6}, 10^{0})$
λ_{FT}	loguniform$(10^{-6}, 10^{0})$
Norm	$\{\ell_1, \ell_{1,\infty}, \ell_{\infty,1}\}$

F Ablations on DomainBed

Comparing the choices for ϕ : In Table 5, we experiment with various architectures for the decoder ϕ when computing FRR according to equation 1. We consider using a two layer neural network as the decoder ϕ (FRR-LDeeper), and also consider setting $\phi = W^T$ (FRR-LShared), i.e. explicitly tying the weights of the decoder and the classifier layer. Overall, both these variants are worse than the default single layer, free parameterization of ϕ. We believe that this happens because the latter approach enforces a much stricter constraint on W, leading to poorer in-domain accuracy, while the former approach enforces a weaker constraint, potentially enabling reconstruction of more complex features from a smaller amount of information about them in the logits. Both these have a detrimental effect on the overall performance of the model.

Table 5: Effect of different design choices on OOD accuracy:the rows shows different architecture choices for ϕ

Algorithm	PACS	OfficeHome	TerraIncognita	**Avg.**
ERM	85.5 ± 0.1	66.5 ± 0.2	46.1 ± 0.6	65.3
ERM+FFR-LShared	85.2 ± 0.5	68.2 ± 0.1	49.4 ± 0.5	67.6
ERM+FFR-LDeeper	84.6 ± 0.7	65.6 ± 0.2	52.5 ± 0.5	67.6

G Domain wise accuracies

In this section, we show detailed results of Table 3 in the main text.

Table 6: **Out-of-domain accuracies (%) on** PACS**.**

Algorithm	**A**	**C**	**P**	**S**	**Avg**
CDANN	84.6 ± 1.8	75.5 ± 0.9	96.8 ± 0.3	73.5 ± 0.6	82.6
MASF	82.9	80.5	95.0	72.3	82.7
DMG	82.6	78.1	94.5	78.3	83.4
IRM	84.8 ± 1.3	76.4 ± 1.1	96.7 ± 0.6	76.1 ± 1.0	83.5
MetaReg	87.2	79.2	97.6	70.3	83.6
DANN	86.4 ± 0.8	77.4 ± 0.8	97.3 ± 0.4	73.5 ± 2.3	83.7
ERM	85.7 ± 0.6	77.1 ± 0.8	97.4 ± 0.4	76.6 ± 0.7	84.2
GroupDRO	83.5 ± 0.9	79.1 ± 0.6	96.7 ± 0.3	78.3 ± 2.0	84.4
MTL	87.5 ± 0.8	77.1 ± 0.5	96.4 ± 0.8	77.3 ± 1.8	84.6
I-Mixup	86.1 ± 0.5	78.9 ± 0.8	97.6 ± 0.1	75.8 ± 1.8	84.6
MMD	86.1 ± 1.4	79.4 ± 0.9	96.6 ± 0.2	76.5 ± 0.5	84.7
VREx	86.0 ± 1.6	79.1 ± 0.6	96.9 ± 0.5	77.7 ± 1.7	84.9
MLDG	85.5 ± 1.4	80.1 ± 1.7	97.4 ± 0.3	76.6 ± 1.1	84.9
ARM	86.8 ± 0.6	76.8 ± 0.5	97.4 ± 0.3	79.3 ± 1.2	85.1
RSC	85.4 ± 0.8	79.7 ± 1.8	97.6 ± 0.3	78.2 ± 1.2	85.2
Mixstyle	86.8 ± 0.5	79.0 ± 1.4	96.6 ± 0.1	78.5 ± 2.3	85.2
ER	87.5	79.3	98.3	76.3	85.3
pAdaIN	85.8	81.1	97.2	77.4	85.4
ERM	84.7 ± 0.4	80.8 ± 0.6	97.2 ± 0.3	79.3 ± 1.0	85.5
EISNet	86.6	81.5	97.1	78.1	85.8
CORAL	88.3 ± 0.2	80.0 ± 0.5	97.5 ± 0.3	78.8 ± 1.3	86.2
SagNet	87.4 ± 1.0	80.7 ± 0.6	97.1 ± 0.1	80.0 ± 0.4	86.3
DSON	87.0	80.6	96.0	82.9	86.6
SMA	93.8 ± 0.3	95.8 ± 0.2	99.2 ± 0.2	93.4 ± 0.2	95.5
SWAD	89.3 ± 0.2	83.4 ± 0.6	97.3 ± 0.3	82.5 ± 0.5	88.1
SWAD+FRR	89.9 ± 0.2	83.9 ± 0.7	98.2 ± 0.3	84.8 ± 0.4	89.2

Table 7: **Out-of-domain accuracies (%) on** VLCS**.**

Algorithm	**C**	**L**	**S**	**V**	**Avg**
GroupDRO	97.3 ± 0.3	63.4 ± 0.9	69.5 ± 0.8	76.7 ± 0.7	76.7
RSC	97.9 ± 0.1	62.5 ± 0.7	72.3 ± 1.2	75.6 ± 0.8	77.1
MLDG	97.4 ± 0.2	65.2 ± 0.7	71.0 ± 1.4	75.3 ± 1.0	77.2
MTL	97.8 ± 0.4	64.3 ± 0.3	71.5 ± 0.7	75.3 ± 1.7	77.2
ERM	98.0 ± 0.3	64.7 ± 1.2	71.4 ± 1.2	75.2 ± 1.6	77.3
I-Mixup	98.3 ± 0.6	64.8 ± 1.0	72.1 ± 0.5	74.3 ± 0.8	77.4
ERM	97.7 ± 0.4	64.3 ± 0.9	73.4 ± 0.5	74.6 ± 1.3	77.5
MMD	97.7 ± 0.1	64.0 ± 1.1	72.8 ± 0.2	75.3 ± 3.3	77.5
CDANN	97.1 ± 0.3	65.1 ± 1.2	70.7 ± 0.8	77.1 ± 1.5	77.5
ARM	98.7 ± 0.2	63.6 ± 0.7	71.3 ± 1.2	76.7 ± 0.6	77.6
SagNet	97.9 ± 0.4	64.5 ± 0.5	71.4 ± 1.3	77.5 ± 0.5	77.8
Mixstyle	98.6 ± 0.3	64.5 ± 1.1	72.6 ± 0.5	75.7 ± 1.7	77.9
VREx	98.4 ± 0.3	64.4 ± 1.4	74.1 ± 0.4	76.2 ± 1.3	78.3
IRM	98.6 ± 0.1	64.9 ± 0.9	73.4 ± 0.6	77.3 ± 0.9	78.6
DANN	99.0 ± 0.3	65.1 ± 1.4	73.1 ± 0.3	77.2 ± 0.6	78.6
CORAL	98.3 ± 0.1	66.1 ± 1.2	73.4 ± 0.3	77.5 ± 1.2	78.8
SMA	98.1 ± 0.2	65.7 ± 1.1	79.2 ± 1.1	79.6 ± 0.2	80.7
SWAD	98.8 ± 0.1	63.3 ± 0.3	75.3 ± 0.5	79.2 ± 0.6	79.1
SWAD+FRR	98.9 ± 0.4	66.3 ± 0.2	75.9 ± 0.6	79.0 ± 0.2	80.0

Table 8: **Out-of-domain accuracies (%) on** `OfficeHome`**.**

Algorithm	**A**	**C**	**P**	**R**	**Avg**
Mixstyle	51.1 ± 0.3	53.2 ± 0.4	68.2 ± 0.7	69.2 ± 0.6	60.4
IRM	58.9 ± 2.3	52.2 ± 1.6	72.1 ± 2.9	74.0 ± 2.5	64.3
ARM	58.9 ± 0.8	51.0 ± 0.5	74.1 ± 0.1	75.2 ± 0.3	64.8
RSC	60.7 ± 1.4	51.4 ± 0.3	74.8 ± 1.1	75.1 ± 1.3	65.5
CDANN	61.5 ± 1.4	50.4 ± 2.4	74.4 ± 0.9	76.6 ± 0.8	65.7
DANN	59.9 ± 1.3	53.0 ± 0.3	73.6 ± 0.7	76.9 ± 0.5	65.9
GroupDRO	60.4 ± 0.7	52.7 ± 1.0	75.0 ± 0.7	76.0 ± 0.7	66.0
MMD	60.4 ± 0.2	53.3 ± 0.3	74.3 ± 0.1	77.4 ± 0.6	66.4
MTL	61.5 ± 0.7	52.4 ± 0.6	74.9 ± 0.4	76.8 ± 0.4	66.4
VREx	60.7 ± 0.9	53.0 ± 0.9	75.3 ± 0.1	76.6 ± 0.5	66.4
ERM	61.3 ± 0.7	52.4 ± 0.3	75.8 ± 0.1	76.6 ± 0.3	66.5
MLDG	61.5 ± 0.9	53.2 ± 0.6	75.0 ± 1.2	77.5 ± 0.4	66.8
ERM	63.1 ± 0.3	51.9 ± 0.4	77.2 ± 0.5	78.1 ± 0.2	67.6
I-Mixup	62.4 ± 0.8	54.8 ± 0.6	76.9 ± 0.3	78.3 ± 0.2	68.1
SagNet	63.4 ± 0.2	54.8 ± 0.4	75.8 ± 0.4	78.3 ± 0.3	68.1
CORAL	65.3 ± 0.4	54.4 ± 0.5	76.5 ± 0.1	78.4 ± 0.5	68.7
SMA	81.1 ± 0.4	72.3 ± 0.6	86.6 ± 0.1	88.2 ± 0.1	82.0
SWAD	66.1 ± 0.4	57.7 ± 0.4	78.4 ± 0.1	80.2 ± 0.2	70.6
SWAD+FRR	65.2 ± 0.2	57.7 ± 0.5	78.2 ± 0.2	80.2 ± 0.1	70.3

Table 9: **Out-of-domain accuracies (%) on** `TerraIncognita`**.**

Algorithm	**L100**	**L38**	**L43**	**L46**	**Avg**
MMD	41.9 ± 3.0	34.8 ± 1.0	57.0 ± 1.9	35.2 ± 1.8	42.2
GroupDRO	41.2 ± 0.7	38.6 ± 2.1	56.7 ± 0.9	36.4 ± 2.1	43.2
Mixstyle	54.3 ± 1.1	34.1 ± 1.1	55.9 ± 1.1	31.7 ± 2.1	44.0
ARM	49.3 ± 0.7	38.3 ± 2.4	55.8 ± 0.8	38.7 ± 1.3	45.5
MTL	49.3 ± 1.2	39.6 ± 6.3	55.6 ± 1.1	37.8 ± 0.8	45.6
CDANN	47.0 ± 1.9	41.3 ± 4.8	54.9 ± 1.7	39.8 ± 2.3	45.8
ERM	49.8 ± 4.4	42.1 ± 1.4	56.9 ± 1.8	35.7 ± 3.9	46.1
VREx	48.2 ± 4.3	41.7 ± 1.3	56.8 ± 0.8	38.7 ± 3.1	46.4
RSC	50.2 ± 2.2	39.2 ± 1.4	56.3 ± 1.4	40.8 ± 0.6	46.6
DANN	51.1 ± 3.5	40.6 ± 0.6	57.4 ± 0.5	37.7 ± 1.8	46.7
IRM	54.6 ± 1.3	39.8 ± 1.9	56.2 ± 1.8	39.6 ± 0.8	47.6
CORAL	51.6 ± 2.4	42.2 ± 1.0	57.0 ± 1.0	39.8 ± 2.9	47.7
MLDG	54.2 ± 3.0	44.3 ± 1.1	55.6 ± 0.3	36.9 ± 2.2	47.8
I-Mixup	59.6 ± 2.0	42.2 ± 1.4	55.9 ± 0.8	33.9 ± 1.4	47.9
SagNet	53.0 ± 2.9	43.0 ± 2.5	57.9 ± 0.6	40.4 ± 1.3	48.6
ERM	54.3 ± 0.4	42.5 ± 0.7	55.6 ± 0.3	38.8 ± 2.5	47.8
SMA	72.4 ± 0.0	52.0 ± 0.5	66.8 ± 0.4	47.4 ± 0.2	59.7
SWAD	55.4 ± 0.0	44.9 ± 1.1	59.7 ± 0.4	39.9 ± 0.2	50.0
SWAD+FRR	60.13 ± 1.05	47.89 ± 1.71	60.76 ± 0.42	42.34 ± 1.35	53.2

Table 10: **Out-of-domain accuracies (%) on** `DomainNet`**.**

Algorithm	**clip**	**info**	**paint**	**quick**	**real**	**sketch**	**Avg**
MMD	32.1 ± 13.3	11.0 ± 4.6	26.8 ± 11.3	8.7 ± 2.1	32.7 ± 13.8	28.9 ± 11.9	23.4
GroupDRO	47.2 ± 0.5	17.5 ± 0.4	33.8 ± 0.5	9.3 ± 0.3	51.6 ± 0.4	40.1 ± 0.6	33.3
VREx	47.3 ± 3.5	16.0 ± 1.5	35.8 ± 4.6	10.9 ± 0.3	49.6 ± 4.9	42.0 ± 3.0	33.6
IRM	48.5 ± 2.8	15.0 ± 1.5	38.3 ± 4.3	10.9 ± 0.5	48.2 ± 5.2	42.3 ± 3.1	33.9
Mixstyle	51.9 ± 0.4	13.3 ± 0.2	37.0 ± 0.5	12.3 ± 0.1	46.1 ± 0.3	43.4 ± 0.4	34.0
ARM	49.7 ± 0.3	16.3 ± 0.5	40.9 ± 1.1	9.4 ± 0.1	53.4 ± 0.4	43.5 ± 0.4	35.5
CDANN	54.6 ± 0.4	17.3 ± 0.1	43.7 ± 0.9	12.1 ± 0.7	56.2 ± 0.4	45.9 ± 0.5	38.3
DANN	53.1 ± 0.2	18.3 ± 0.1	44.2 ± 0.7	11.8 ± 0.1	55.5 ± 0.4	46.8 ± 0.6	38.3
RSC	55.0 ± 1.2	18.3 ± 0.5	44.4 ± 0.6	12.2 ± 0.2	55.7 ± 0.7	47.8 ± 0.9	38.9
I-Mixup	55.7 ± 0.3	18.5 ± 0.5	44.3 ± 0.5	12.5 ± 0.4	55.8 ± 0.3	48.2 ± 0.5	39.2
SagNet	57.7 ± 0.3	19.0 ± 0.2	45.3 ± 0.3	12.7 ± 0.5	58.1 ± 0.5	48.8 ± 0.2	40.3
MTL	57.9 ± 0.5	18.5 ± 0.4	46.0 ± 0.1	12.5 ± 0.1	59.5 ± 0.3	49.2 ± 0.1	40.6
ERM	58.1 ± 0.3	18.8 ± 0.3	46.7 ± 0.3	12.2 ± 0.4	59.6 ± 0.1	49.8 ± 0.4	40.9
MLDG	59.1 ± 0.2	19.1 ± 0.3	45.8 ± 0.7	13.4 ± 0.3	59.6 ± 0.2	50.2 ± 0.4	41.2
CORAL	59.2 ± 0.1	19.7 ± 0.2	46.6 ± 0.3	13.4 ± 0.4	59.8 ± 0.2	50.1 ± 0.6	41.5
MetaReg	59.8	25.6	50.2	11.5	64.6	50.1	43.6
DMG	65.2	22.2	50.0	15.7	59.6	49.0	43.6
ERM	63.0 ± 0.2	21.2 ± 0.2	50.1 ± 0.4	13.9 ± 0.5	63.7 ± 0.2	52.0 ± 0.5	44.0
SMA	78.8 ± 0.1	43.0 ± 0.0	68.6 ± 0.0	21.2 ± 0.0	78.5 ± 0.1	69.8 ± 0.1	60
SWAD	66.0 ± 0.1	22.4 0.3	53.5 ± 0.1	16.1 ± 0.2	65.8 ± 0.4	55.5 ± 0.3	46.5
SWAD+FRR	65.9 ± 0.1	22.3 ± 0.0	52.8 ± 0.1	14.8 ± 0.3	66.2 ± 0.1	55.0 ± 0.1	46.2