The Two Dimensions of Worst-case Training and Their Integrated Effect for Out-of-domain Generalization

Zeyi Huang\(^1\)\(^,*\), Haohan Wang\(^1\)\(^,*\), Dong Huang\(^1\), Yong Jae Lee\(^2\)\(^†\), Eric P. Xing\(^1\)\(^†\)

\(^1\)Carnegie Mellon University \(^2\)University of Wisconsin-Madison
{zeyih@andrew, haohanw@cs, donghuang@, epxing@cs}.cmu.edu yongjaeelee@cs.wisc.edu

Abstract

Training with an emphasis on “hard-to-learn” components of the data has been proven as an effective method to improve the generalization of machine learning models, especially in the settings where robustness (e.g., generalization across distributions) is valued. Existing literature discussing this “hard-to-learn” concept are mainly expanded either along the dimension of the samples or the dimension of the features. In this paper, we aim to introduce a simple view merging these two dimensions, leading to a new, simple yet effective, heuristic to train machine learning models by emphasizing the worst-cases on both the sample and the feature dimensions. We name our method W2D following the concept of “Worst-case along Two Dimensions”. We validate the idea and demonstrate its empirical strength over standard benchmarks.

1. Introduction

The remarkable empirical performance of deep learning over i.i.d data, sometimes paralleling the human visual system [23, 37], has encouraged the community to challenge potentially more demanding scenarios where the models are trained with data from one or more distributions but tested with data from other distributions. We refer to this scenario as the out-of-distribution (OOD) generalization testing setting following the terminology used in [74].

In this OOD test scenario, deep learning techniques often underdeliver the promising results made with i.i.d data, as observed by multiple preceding works with different strategies to generate the test data, such as with salient patterns added to the data [17, 27], with carefully constructed imperceptible noise perturbing the data (adversarial attacks) [20, 60], or with additionally collected datasets that humans can nonetheless generalize to despite potentially significant disparities between the training and test distributions (e.g., domain adaptation/generalization) [6, 48].

Despite the variation in multiple OOD settings, the underlying reason leading to the performance drop may have a shared theme: the models’ incapability to learn what humans will consider important in the data, as discussed previously in empirical [70] and statistical [68] perspectives.

Thus, we conjecture that a key to training models that can perform consistently well in the OOD setting is to design a new training heuristic that can better imitate the hu-
man’s learning behaviors. In addition, we also hope the new heuristic is simple and general so that it can be directly plugged into and benefit existing methods across different architectures, optimizers, losses, or regularizations.

A psychological prior: In seeking the answer of how a human can learn most efficiently, we notice that a world-renowned psychologist (Dr. K. Anders Ericsson) has devoted his life-long career decoding the habits of people with expert-level performances. His main conclusion [14] is that the high-end performances are the result of extensive practices beyond one’s comfort zone.

Back to the discussion of machine learning, we analogize the beyond comfort zone elements of one’s daily life to the elements of the training data that are particularly hard to learn for a model. We notice this “element” can be interpreted with two perspectives: one interpretation is that a certain pattern across many images is hard to learn; and the other is that some specific images in the dataset are hard to learn.

Previous discussions devoted exclusively to either one of these two elements has been expanded extensively. For example, a line of methods have been invented to counter the model’s tendency to learn some simple patterns [4, 50, 65, 66], and another line of methods have been introduced to push the model to learn patterns represented by a small set of samples [38, 57]. However, there seems to be no discussion aiming to train the model to overcome the limitations raised from both of these perspectives, while doing so would intuitively improve the model’s performances, as well as align well with the psychological findings mentioned above.

In this paper, inspired by the psychological prior above, we aim to introduce a simple training heuristic that will push the model to learn the hard-to-learn concept on both the feature dimension and the sample dimension. As our method is intuitively a combination of worst-case training at the feature level and worst-case training at the sample level, the new technique can serve as a simple heuristic to replace the hard-to-learn elements from the data, we continue the discussion with a focus on the worst-case training methods along both dimensions. Finally, we wrap up this section with a summary of the key contributions made in this paper.

2. Background

We introduce the background of our work in this section. We first offer a brief summary of works that improve a model’s OOD generalization performance. We then focus on related work that are devoted to solving the two challenges at the feature level and at the sample level, respectively. As we notice that some of the methods solving these two problems have a common theme of emphasizing the hard-to-learn elements from the data, we continue the discussion with a focus on the worst-case training methods along both dimensions. Finally, we wrap up this section with a summary of the key contributions made in this paper.

2.1. Domain Adaptation, Domain Generalization, and New Paradigms

The investigation of a model’s generalization ability across distributions can probably be traced back to the study of domain adaptation [6, 7], which studies the general problem of maintaining a model’s performance over a test distribution that is different from the training distribution. Early-stage theoretical work suggests one of the key factors of learning cross-domain generalizable models is to enforce invariance across distributions [6], and this has inspired a long line of work aiming to learn invariant representations across the training and testing distributions [13, 15, 18, 34, 44, 56, 71, 77], with the most popular recent examples being domain adversarial neural networks [16].

Domain generalization [48] is another mainstream research topic in the study of OOD generalization. It extends the setup of domain adaptation to a setting in which the testing distribution data, even unlabelled, is not available during training. Instead, models are trained with data from multiple training distributions, and enforcing invariance across these training distributions has become a major theme [1, 10, 19, 22, 42, 47, 52, 54, 69].

However, recently, the paradigm of learning invariant representations has been challenged by the argument that invariance is not sufficient for cross-domain generalization if the data have different labelling functions [72, 78], potentially leading to a new paradigm of learning across distributions with disparate labelling functions [68], as will be detailed with examples in the next section.

As a result, recent studies are not always bounded by the concepts of domain adaptation or domain generalization, but are conducted in the new paradigm with disparate labelling functions.

2.2. Hard-to-learn Patterns and Solutions

One of the mainstream studies in robust machine learning focuses on the issue of models learning some patterns in the training data that are not present in the test data, with the most popular example probably being the snow background in husky vs. wolf classification [55]. This problem is of-
en referred to as biases [61], spurious features [64], confounding factors [45], or superficial features [66], but the solution to counter the problem usually has a unified theme of leveraging the human knowledge of the differences between training and testing distributions to either regularize the hypothesis space [4, 50, 65, 66] or to augment the data [17, 28, 29, 67], as summarized in [68].

Interestingly, the RSC method [31] also aims to solve the challenge along this line, but it does not require prior knowledge over the patterns. Building upon an assumption that learning all the features, instead of just the most discriminative ones, will benefit the OOD generalization, RSC essentially uses a selective dropout mechanism to perform augmentation, and achieves good benchmark performances on popular OOD datasets [31, 74]. Conceptually, RSC prepares the features for each sample by dropping out the most predictive features (i.e., creating features that are challenging for the model to learn).

### 2.3. Hard-to-learn Samples and Solutions

On the other hand, hard-to-learn samples pose a different challenge: some samples in the training data are ignored by the model because these samples are considered to be the “minority” in the training set [12, 46]. To counter this problem, training procedures that emphasize the minority samples have been introduced, such as the family of DRO methods [30, 39, 50, 51, 57], with different strategies to identify the minority samples and to interpolate the training set according to weighting factors that favor the minority samples. Intuitively, these methods prepare the batches of samples with an emphasis on the samples that are challenging for the model to learn.

Further along this line, the community extends the idea of interpolation to extrapolation by adjusting the weighting factors so that the weights for simpler samples can even be negative, to further push the models to focus on the hard-to-learn samples. The VREx method [38] is introduced in this context and achieves leading empirical performances on benchmarks [38, 74].

### 2.4. Worst-case Training in Each Dimension and the Corresponding Effects

With the proliferation of methods introduced to improve models’ OOD performances, and various empirical claims, there have been some integrated comparisons of the various methods. For example, interestingly, DomainBed suggests that all these new methods are still shy of the conventional Empirical Risk Minimization (ERM) method [21] under their extensive range of hyperparameter choices. While this message sets a striking alarm to the community, it may seem overly pessimistic (more details on this are offered later in Section 6).

Recently, the OOD-bench [74] extends the spirit of DomainBed but offers a more finer-grained analysis of the models’ performances. It comprehensively examines the performances of the recent models over popular benchmark datasets, but with separate discussions on “diversity shift” datasets and “correlation shift” datasets. Diversity shift datasets refer to the benchmarks with a relatively significant style shift from training distribution to the test distribution (such as from photos to sketches), and correlation shift datasets refer to the benchmarks with clear defined spurious features that are correlated with the label (such as when the color of the digits are associated with the labels of the digits in the image digit classification task).

OOD-bench investigates the popular methods along these two directions, and shows that for each direction, there are only a couple of methods that outperform ERM. Interestingly, the best performing method for diversity shift is RSC [31], one of the most advanced methods aimed to learn hard-to-learn patterns, with a heuristic to push the model to train with generated worst-case features. On the other hand, the best performing method for correlation shift is VREx [38], one of the most advanced methods aimed to learn hard-to-learn samples, with a heuristic to push the model to train with selected worst-case samples.

Our contribution is inspired by the above discussion: if worst-case training in the feature dimension excels at diversity shift, while worst-case training in the sample dimension excels at correlation shift, if we can integrate these two worst-case training methods into one simple heuristic, the new method will likely lead to sufficiently good empirical performance for both diversity and correlation shift.

### 2.5. Key Contributions

In comparison to the previous methods discussed along these two lines, we believe the key contributions we make in this paper are as follows.

- We discuss the previous methods with a unified theme of worst-case training along two different dimensions, and doing so naturally leads to the integration of these methods.
- We introduce a new method, called W2D, as an integration of these two types of training methods. W2D is a simple heuristic that can be directly plugged into any training process regardless of model architecture, loss function, regularization or optimizer.
- We demonstrate strong empirical performance on multiple benchmark datasets, and conduct ablation studies to understand the contribution of each of component of the method.

### 3. Method

In this section, we first formalize the two worst-case training methods, which naturally leads to the introduction of our method. Then, with the main framework of our
method introduced, we continue to discuss a whole-batch patching heuristic we use in the experiments that have benefited our method empirically with non-negligible margins.

3.1. W2D Method

We first introduce our notations. We use \( (X, Y) \) to denote a dataset with \( n \) (data, label) paired samples. Thus \( X \in \mathbb{R}^{n \times p} \) and \( Y \in \mathbb{R}^n \). We use \( f(\cdot; \theta) \) to denote the model we aim to train, and use \( e(\cdot; \theta_e) \) and \( d(\cdot; \theta_d) \) to denote encoder and decoder, respectively. Thus, we have \( f(\cdot; \theta) = d(e(\cdot; \theta_e); \theta_d) \). We use \( w \) to denote a weight vector of length \( n \). We use \( m \) to denote a masking vector with some elements to be 0 and others to be 1; the length of \( m \) is the same as the feature dimension (the output of \( e(\cdot; \theta_e) \)). We use \( l(\cdot, \cdot) \) to denote a generic loss function.

The vanilla training process of a model is

\[
\hat{\theta}_{\text{vanilla}} = \arg \min_{\theta} \frac{1}{n} \sum_{i} l(f(X_i; \theta), Y_i)
\]

Worst-case along feature dimension We formalize the first worst-case method, with a generic form as follows:

\[
\hat{\theta}_{\text{feature}} = \arg \min_{\theta} \frac{1}{n} \sum_{i} \max_{m} l(d(m \odot e(X_i; \theta_e); \theta_d), Y_i),
\]

subject to \( \sum_{i} w_i = 1 \)

where \( \odot \) denotes the element-wise product.

In particular, the RSC method [31] introduces the \( m \) with a hyperparameter \( \rho \), which denotes the fraction of the elements that are zeros. The maximization step is achieved by examining the magnitude of the gradient of \( \frac{\partial d(e(\cdot; \theta_e); \theta_d)}{\partial e} \).

Worst-case along sample dimension We formalize the second worst-case method, with a generic form as follows:

\[
\hat{\theta}_{\text{sample}} = \arg \min_{\theta} \frac{1}{n} \sum_{i} \max_{w_i} w_i l(f(X_i; \theta), Y_i),
\]

subject to \( \sum_{i} w_i = 1 \)

In general, the choice of \( w_i \) depends on \( l(f(X_i; \theta), Y_i) \), with concrete differences across different methods such as [39, 50, 57]. A common theme is that, the higher the loss is, the bigger \( w_i \) is.

In practice, because we use batch-wise optimization, the estimation of \( w \) is not straightforward. Fortunately, we can use a simple alternative: for each batch, we select the samples with high losses (through a forward pass), and then use these samples to update the model. This is a heuristic used by multiple methods such as [9, 11, 33, 73].

### Algorithm 1: W2D Algorithm

| Input: | data set \((X, Y)\), percentage of samples used per batch \( \rho \), percentage of whole batch patching \( \kappa \), batch size \( \eta \), maximum number of epochs \( T \), and other RSC hyperparameters; |
| Output: | Classifier \( f(\cdot; \theta) \); |
| randomly initialize the model \( \theta_0 \); |
| calculate the number of iterations \( K = n/\eta \); |
| while \( t \leq (1 - \kappa)T \) do |
| for a batch of data \((X, Y)_k\) where \( k \leq K \) do |
| forward pass to calculate the loss \( l(f(X_i; \theta_{t,k-1}), Y_i) \) of every sample in the batch; |
| select the top \( \eta \rho \) samples with highest loss to construct \((X, Y)_{k,\rho'}\); |
| Train the model with \((X, Y)_{k,\rho'}\) following (1). |
| end |
| end |
| while \((1 - \kappa)T < t \leq T \) do |
| for a batch of data \((X, Y)_k\) where \( k \leq K \) do |
| Train the model with \((X, Y)_k\) following (1). |
| end |

W2D Method Integrating the two methods above, we have

\[
\hat{\theta}_{\text{W2D}} = \arg \min_{\theta} \frac{1}{n} \sum_{i} \max_{m, w_i} w_i l(d(m \odot e(X_i; \theta_e); \theta_d), Y_i),
\]

subject to \( \sum_{i} w_i = 1 \)

In practice, we use the RSC method to identify the \( m \) for worst-case training in the feature dimension, and use the above heuristic to perform worst-case training in the sample dimension.

3.2. Whole-batch Patching Heuristic

As introduced above, W2D selects the worst-case training samples with highest loss in each training iteration, as the model evolves over training, it is possible that what was once considered easy becomes hard, and vice versa. Therefore, we can intuitively expect the model to see all the samples in the training set given sufficient training iterations. However, chances are that some of the samples are never seen by the model during training as these samples are always considered not hard enough; it is obvious not ideal that there is a chance the model does not take full advantage of the training set.

To counter this potential issue, we simply switch to whole batch training during the last \( \kappa \% \) of training epochs,
which leads to better empirical performances across different model selection strategies. We verify the effectiveness of this simple approach in our ablation study. More results can be found in Section 5.4.

The full description of W2D with the whole batch patching heuristic is detailed in Algorithm 1.

4. Experiments

4.1. Experimental Setup

We follow the setting in [74] and evaluate domain generalization on both types of distribution shift: diversity shift and correlation shift. Specifically, we use the same strategy for model selection, dataset splitting, and network backbone. More details of the experimental settings can be found in the discussion and supplementary materials.

4.2. Datasets, Hyperparameter Search and Model Selection

We choose datasets that cover as much variety as possible from the various OoD research areas for our experiments. We conduct experiments on seven OOD datasets: CMNIST [2], CelebA [43], NICO [25], Terra Incognita [5], OfficeHome [63], WILDS-Camelyon, [35] and PACS [40]. These datasets are divided into two categories based on their estimated diversity and correlation shift.

We use the same hyperparameter search protocol as [21, 74]: a 20-times random hyperparameter search is conducted for every dataset and algorithm pair, and then the search process is repeated for another two random series of hyperparameter combinations, weight initializations, and dataset splits. The three series yield the three best accuracies in total over which a mean and standard error bar is computed for every dataset-algorithm pair.

To be consistent with existing line of work, models trained on PACS, OfficeHome, and Terra Incognita are selected by training-domain validation; models trained on WILDS-Camelyon and NICO are selected by leave-one-domain-out validation; while models trained on Colored MNIST and CelebA are selected by test-domain validation. Details of these selection strategies can be found in [74].

4.3. Empirical Results

The benchmark results are shown in Table 1 and Table 2. In addition to mean accuracy and standard error bar, we follow the Ood-bench [74] to report a ranking score for each algorithm with respect to Empirical Risk Minimization (ERM) [62]. Specifically, depending on whether the attained accuracy is lower than, within, or higher than the standard error bar of ERM accuracy on the same dataset, scores -1, 0, +1 are assigned to every dataset-algorithm pair. Adding up the scores across all datasets listed in the table produces the ranking score for each algorithm. The ranking score reflects a relative degree of robustness against diversity and correlation shift compared to ERM.

Note, for CMNIST, Ood-bench [74] evaluates the results using the -90 as testing domain while DomainBed [21] reports the results averaged over the +90, +80, and -90 domains. We follow Ood-bench’s setting to report the results in Table 2 and we also report the results of the DomainBed’s setting in the supplementary material. The choice of the settings does not affects our ranking score in Table 2. In Section 5, we discuss a special property of CMNIST and propose a modified version of W2D which can achieve a significant gain on CMNIST.

We observe that W2D is the only algorithm that can achieve consistently better performance than ERM on both types of distribution shifts. Specifically, W2D is among the top three in both the datasets dominated by diversity shift as well as the datasets dominated by correlation shift. This comprehensive evaluation supports the view that W2D could serve as a simple heuristic to replace existing training approaches for real-world applications because real-world data have both kinds of distribution shifts.

5. Ablation Study

There are altogether four hyperparameters for W2D, two of them are directly inherited from RSC [31]: feature dropping percentage, \( \phi \), controls the different dropping percentages to mute feature maps; batch dropping percentage, \( \beta \), controls the different batch size percentages to apply feature dropping. There are two new hyperparameters introduced along the sample dimension by W2D: worse-case sample percentage, \( \rho \), controls the fraction of the batch size samples with the highest loss used for training; whole batch patching percentage, \( \kappa \), controls the percentage of training time trained using the whole batch.

We use the default hyperparameters from RSC for feature dropping percentage and batch dropping percentage. For selecting the worst-case along the sample dimension, we conduct two ablation studies on possible configurations for it on the standard benchmarks. All results are produced by following Ood-bench’s setting.

Overall, we set the hyperparameter search space of W2D as \( \phi \in [0.1, 0.4] \), \( \beta \in [0.1, 0.3] \), \( \rho \in [0.1, 0.5] \), \( \kappa \in [0.2, 0.4] \).

5.1. Effect of Worst-case sample percentage \( \rho \)

We test W2D with different percentages of worst-case batch samples in Table 3. For PACS, the fewer the worst-case samples used for training, the higher the test-validation accuracy. This result suggests that focusing on more hard-to-learn worse-case samples can better push the limit of the model’s potential generalization power as indicated by the higher test-validation accuracy.
In Table 4, we vary $\kappa$: 0 percent means never training with the whole batch. As we increase $\kappa$, we observe higher training-validation accuracy, but lower testing-validation accuracy. This ablation study demonstrates that whole batch training can boost training validation results, while slightly decreasing the model’s potential generalization ability at the same time.

5.3. Dimensions of Worst-case Training

In Table 5, we evaluate each component of W2D. Both components (sample dimension and feature dimension) are shown to outperform ERM. We believe each component can be easily plugged into other domain generalization methods and achieve consistent gains. Also, integrating both components is the best setting (W2D) for most of the diversity shift and correlation shift datasets.
### Table 3. Ablation study of Top Worst-case ρ%. We fix other hyperparameters here and only change ρ.

<table>
<thead>
<tr>
<th>Percentage</th>
<th>Dataset</th>
<th>Acc(Train-Val/Test-Val)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>PACS</td>
<td>82.4 / 83.7</td>
</tr>
<tr>
<td>20</td>
<td>PACS</td>
<td>83.0 / 83.5</td>
</tr>
<tr>
<td>33</td>
<td>PACS</td>
<td>82.7 / 83.2</td>
</tr>
<tr>
<td>50</td>
<td>PACS</td>
<td>82.7 / 83.1</td>
</tr>
</tbody>
</table>

### Table 4. Ablation study of Whole Batch Training During The Last κ% Epoch. We fix other hyperparameters here and only change κ.

<table>
<thead>
<tr>
<th>Percentage</th>
<th>Dataset</th>
<th>Acc(Train-Val/Test-Val)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>PACS</td>
<td>82.2 / 83.7</td>
</tr>
<tr>
<td>5</td>
<td>PACS</td>
<td>82.5 / 83.5</td>
</tr>
<tr>
<td>10</td>
<td>PACS</td>
<td>82.7 / 83.3</td>
</tr>
<tr>
<td>20</td>
<td>PACS</td>
<td>83.0 / 83.3</td>
</tr>
<tr>
<td>40</td>
<td>PACS</td>
<td>82.9 / 83.3</td>
</tr>
</tbody>
</table>

### Table 5. Analysis of each dimension of W2D.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Dataset</th>
<th>Acc(Train-Val/Test-Val)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERM</td>
<td>PACS</td>
<td>81.5 / 82.2</td>
</tr>
<tr>
<td>feature-dim.</td>
<td>PACS</td>
<td>82.8 / 83.3</td>
</tr>
<tr>
<td>sample-dim.</td>
<td>PACS</td>
<td>82.2 / 83.5</td>
</tr>
<tr>
<td>W2D</td>
<td>PACS</td>
<td>83.4 / 84.0</td>
</tr>
<tr>
<td>ERM</td>
<td>OfficeHome</td>
<td>63.3 / 63.5</td>
</tr>
<tr>
<td>feature-dim.</td>
<td>OfficeHome</td>
<td>62.9 / 63.3</td>
</tr>
<tr>
<td>sample-dim.</td>
<td>OfficeHome</td>
<td>63.3 / 63.7</td>
</tr>
<tr>
<td>W2D</td>
<td>OfficeHome</td>
<td>63.5 / 63.8</td>
</tr>
<tr>
<td>ERM</td>
<td>TerraInc</td>
<td>42.6 / 43.9</td>
</tr>
<tr>
<td>feature-dim.</td>
<td>TerraInc</td>
<td>43.6 / 44.8</td>
</tr>
<tr>
<td>sample-dim.</td>
<td>TerraInc</td>
<td>42.9 / 45.1</td>
</tr>
<tr>
<td>W2D</td>
<td>TerraInc</td>
<td>44.5 / 46.3</td>
</tr>
<tr>
<td>ERM</td>
<td>CelebA</td>
<td>86.3 / 87.2</td>
</tr>
<tr>
<td>feature-dim.</td>
<td>CelebA</td>
<td>86.2 / 85.9</td>
</tr>
<tr>
<td>sample-dim.</td>
<td>CelebA</td>
<td>85.8 / 87.4</td>
</tr>
<tr>
<td>W2D</td>
<td>CelebA</td>
<td>86.5 / 87.7</td>
</tr>
</tbody>
</table>

5.4. Training Validation vs Testing Validation

For training-domain validation, each training domain is split into training and validation subsets. The models are trained using the training subsets and the final model is chosen as the one that maximizes the accuracy on the union of the validation subsets. Training validation is designed to apply on real-world applications. For testing-domain validation, the model is selected by maximizing the accuracy on a validation set that follows the distribution of the test domain. Testing validation is used to measure a method’s highest potential generalization ability. In Table 5, we see that compared to RSC (feature-dim.), W2D tends to obtain bigger improvements when testing-domain validation is used for model selection than training-domain validation. This is mainly because worst-case training along sample dimension can increase the model’s potential generalization power.

5.5. A special property of Colored MNIST

As mentioned earlier, we evaluate the results in CMNIST using the -90 as testing environment in Table 2 following Ood-Bench [74]. In this section, we report the results averaged over three environments (+90, +80 and -90) in CMNIST, which is the protocol used in DomainBed [21]. The study of these results leads us to notice a special property of CMNIST in comparison to other methods used. Then, this special property leads us to introduce a modified version of W2D that improves over ERM by a clear margin by taking advantage of this property.

Treating the -90 domain as a testing domain is considered to be the most difficult setting because the training and testing domain’s distribution are totally flipped. (In comparison, the discussions of the other two testing domains, +90 and +80, are omitted as they are much simpler.) If we can train on only a small subset of the training samples that share the same distribution as that of testing, the results could be largely improved. Worst-case training along the sample dimension would be a natural solution for this problem. However, we found that a vanilla usage of this method can heavily affect training for this toy dataset.

To solve this problem, we first train a biased classifier at the beginning of training with a few epochs. The biased classifier is then fixed and used as a pre-trained classifier to select the worse-case samples. Specifically, we utilize the worse-case samples selected in each iteration by the biased classifier to train a debiased classifier. Since the distribution flips between the training and testing domain (from +80/+90 to -90), the worse-case samples selected by the pre-trained biased classifier should share a similar distribution with the samples during testing, which leads to surprisingly high performance in Table 6.

We hope this ablation study can motivate the community to rethink the evaluation method of Colored MNIST. We conjecture a more reasonable protocol is to, rather than only reporting results on -90, evaluate the methods with multiple different distribution domains: e.g., averaged over +/-90, +/-70, +/-50, +/-30, +/-10 domains.

6. Discussion

Additional Benefit with Stochastic Weight Averaging

Stochastic Weight Averaging (SWA) [32] is an ensemble technique that finds the solution at the center of a wide flat region of the loss landscape. It performs an equal averaging of the model parameters derived from multiple local minima during the training procedure. SWA was shown to im-
Table 6. The CMNST results are adopted from [21] and averaged over three domains. * means modified version of W2D.

<table>
<thead>
<tr>
<th>Method</th>
<th>Dataset</th>
<th>Acc(Train-Val/Test-Val)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERM [62]</td>
<td>CMNST</td>
<td>51.5 / 58.5</td>
</tr>
<tr>
<td>GroupDRO [57]</td>
<td>CMNST</td>
<td>52.1 / 61.2</td>
</tr>
<tr>
<td>VREx [38]</td>
<td>CMNST</td>
<td>51.8 / 56.3</td>
</tr>
<tr>
<td>ARM [76]</td>
<td>CMNST</td>
<td>56.2 / 63.2</td>
</tr>
<tr>
<td>IRM [2]</td>
<td>CMNST</td>
<td>52.0 / 70.2</td>
</tr>
<tr>
<td>RSC [31]</td>
<td>CMNST</td>
<td>51.7 / 58.5</td>
</tr>
<tr>
<td>W2D</td>
<td>CMNST</td>
<td>51.9 / 59.0</td>
</tr>
<tr>
<td>W2D*</td>
<td>CMNST</td>
<td>70.8 / 72.9</td>
</tr>
</tbody>
</table>

Table 7. W2D can further improve the performances if used together with Stochastic Weight Averaging. We apply SWA at the last 25 percent of training time and do not apply whole batch patching here.

<table>
<thead>
<tr>
<th>Method</th>
<th>Dataset</th>
<th>Acc(Train-Val/Test-Val)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERM</td>
<td>PACS</td>
<td>81.5 / 82.2</td>
</tr>
<tr>
<td>feat-dim.</td>
<td>PACS</td>
<td>82.8 / 83.3</td>
</tr>
<tr>
<td>sample-dim.</td>
<td>PACS</td>
<td>82.2 / 83.5</td>
</tr>
<tr>
<td>W2D</td>
<td>PACS</td>
<td>83.4 / 84.0</td>
</tr>
<tr>
<td>ERM(w SWA)</td>
<td>PACS</td>
<td>82.5 / 83.0</td>
</tr>
<tr>
<td>feat-dim.(w SWA)</td>
<td>PACS</td>
<td>83.5 / 83.7</td>
</tr>
<tr>
<td>sam-dim.(w SWA)</td>
<td>PACS</td>
<td>83.4 / 83.7</td>
</tr>
<tr>
<td>W2D(w SWA)</td>
<td>PACS</td>
<td>84.7 / 84.8</td>
</tr>
</tbody>
</table>

Challenges for our method in DomainBed First, methods such as RSC [31] or the family of DRO methods [38,57] are simple heuristic extensions of ERM along the feature or sample dimension. It seems counter-intuitive that these methods cannot compete with ERM if used properly. When closely studying the experimental settings in DomainBed, first, we notice the hyperparameter range of RSC goes as high as dropping 50% of the features, and with such a high dropout rate, we find that RSC can barely learn any useful patterns. Second, DomainBed changes the default model setting of ResNet50 [24] by adding dropout [58] in the fully connected layers. The highest dropout rate in DomainBed is 50%, which might be beneficial to other algorithms but could degrade RSC’s performance due to the overuse of dropout from both aspects. For the other dimension, the batchsize range goes as small as 8, which limits the potential of the DRO-family methods to use the hard samples.

Why Ood-bench? First, Ood-bench uses a ranking score to reflect a relative degree of robustness against both kinds of distribution shift instead of mean accuracy over different datasets, which is more reasonable. Several algorithms are superior to ERM in the toy case datasets, but they are still vulnerable to the distribution shift from the real data. Thus, using mean accuracy is a less meaningful way to compare these algorithms. Second, unlike DomainBed, Ood-bench uses a smaller model, ResNet18 [24], for all algorithms and datasets excluding Colored MNIST. It is known that larger models are usually more robust to distribution shift data and thus their performance may be more easily saturated on small datasets [26]. Thus, using a smaller base model on which to build on could provide a better testbed for OoD generalization of different algorithms. Third, the search space of the non-algorithm-specific hyperparameters is carefully designed, such as the learning rate. It allows each algorithm to converge during training at each run. More importantly, Ood-bench measures each method’s generalization ability in a more objective and fair manner: it excludes previous domain generalization techniques that can be plugged into any algorithm, such as dropout [58].

Limitations In the more realistic dataset dominated by correlation shift, such as CelebA and NICO, although W2D achieves the best improvement among all the algorithms, it does not surpasses ERM by a statistically significant margin. Despite the high empirical performances, there are no sufficient evidence suggesting that W2D will be ideal when facing common challenge such as spurious correlations. Study to extend W2D to further overcome these challenges is likely expected in the future.

7. Conclusion
Inspired by a simple heuristic that training with a particular focus on hard-to-learn concepts will benefit the learning process, in this paper, we introduce a training heuristic method that can iteratively force the model to learn the hard-to-learn concepts on both the feature dimension and the sample dimension. We name our method W2D following the idea of “Worst-case along Two Dimensions”. W2D can be directly applied to almost any model architecture, optimizer, loss, or regularization etc. After evaluating W2D in Ood-Bench comprehensively, we observe W2D is the only algorithm that can achieve consistently better performance than ERM on both diversity shift and correlation shift.

Acknowledgements. This work was supported in part by NSF CAREER IIS-2150012 and IIS-2204808. HW was supported by NIH R01GM114311, NIH P30DA035778, and NSF IIS1617583.
References


9639


[33] Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. In International conference on machine learning, pages 2525–2534. PMLR, 2018. 4


[40] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In Thirty-Second AAAI Conference on Artificial Intelligence, 2018. 6


[53] Mohammad Mahfujur Rahman, Clinton Fookes, and Sridha Sridharan. Discriminative domain-invariant adversarial net-


[56] Kate Saenko, Brian Kulis, Mario Fritz, and T Darrell. Adapting visual category models to new domains. In *Proceedings of the European conference on computer vision*, 2010. 2


