Discovering Environments with XRM

Mohammad Pezeshki [1] **Diane Bouchacourt** [1] **Mark Ibrahim** [1] **Nicolas Ballas** [1]
Pascal Vincent [1 2 3] **David Lopez-Paz** [1]

Abstract

Environment annotations are essential for the success of many out-of-distribution (OOD) generalization methods. Unfortunately, these are costly to obtain and often limited by human annotators' biases. To achieve robust generalization, it is essential to develop algorithms for automatic environment discovery within datasets. Current proposals, which divide examples based on their training error, suffer from one fundamental problem. These methods introduce hyper-parameters and early-stopping criteria, which require a validation set with human-annotated environments, the very information subject to discovery. In this paper, we propose CROSS-RISK MINIMIZATION (XRM) to address this issue. XRM trains twin networks, each learning from one random half of the training data, while imitating confident held-out mistakes made by its sibling. XRM provides a recipe for hyper-parameter tuning, does not require early-stopping, and can discover environments for all training and validation data. Algorithms built on top of XRM environments achieve oracle worst-group-accuracy, addressing a long-standing challenge in OOD generalization. Code available at https://github.com/facebookresearch/XRM.

1 Introduction

AI systems pervade our lives, spanning applications such as finance (Hand & Henley, 1997), healthcare (Jiang et al., 2017), self-driving vehicles (Bojarski et al., 2016), and justice (Angwin et al., 2016). Despite outperforming humans in many tasks, these systems fall apart under testing conditions different from their *training environments* (Geirhos et al., 2020). For instance, during the COVID-19 pandemic, thoracic x-ray classifiers incorrectly latched onto spurious correlations such as patient's age or position (Heaven, 2021), leading to "an alarming situation in which the systems appear accurate, but fail when tested in new hospitals" (DeGrave et al., 2021).

Generally speaking, AI systems underperform on underrepresented groups in training data (Barocas et al., 2019). The Waterbirds problem (Sagawa et al., 2019), depicted in Figure 1, illustrates this with two classes (landbirds and waterbirds) in two landscape environments (land and water), forming four groups: a *majority group* of waterbirds in water (73% of examples), landbirds in land (22%), waterbirds in land (4%), and a *minority group* of landbirds in water (1%). On this problem, learning machines often favor the *landscape* spurious feature to classify the majority of examples and memorizes the remaining minorities to achieve zero training error. An empirical risk minimization (ERM) baseline, ignoring environment data (Vapnik, 1998), achieves a mere 61% worst-group-accuracy, specifically on the minority group, as illustrated in Figure 1's right panel.

To improve upon ERM, researchers have developed a myriad of OOD generalization algorithms (Zhou et al., 2022a; Wang et al., 2021). These methods use environment annotations to uncover invariant (environment-generic) patterns and discard spurious (environment-specific) ones (Arjovsky et al., 2019). As Figure 1 shows, *group distributionally robust optimization* (Sagawa et al., 2019, GroupDRO) achieves a worst-group-accuracy of 87%. This outperforms ERM by over twenty five points, a sizeable gap!

While promising, OOD algorithms targeting sub-population shift require environment annotations, which are costly to obtain and limited by human annotators' biases and precision. Moreover, no single set of environment annotations fits all OOD algorithms. The patterns misleading a learning system might be alien or invisible to humans (Goodfellow et al., 2014). Because of these reasons, OOD generalization is currently confined to small data collections, and their promise in the large-scale setting remains unfulfilled.

In light of the above, researchers developed algorithms for automatic environment discovery (Bao & Barzilay, 2022; Zheran Liu et al., 2021; Zhang et al., 2022b; Lahoti et al., 2020; Dagaev et al., 2021; Creager et al., 2020; Nam et al., 2020). These methods typically build a robust system in two phases. In phase-1, these methods train a classifier to categorize training examples in two environments, based on their training error. In phase-2, an OOD algorithm is

[1]FAIR at Meta [2]Mila at Université de Montréal [3]CIFAR. Correspondence to: Mohammad Pezeshki <mpezeshki@meta.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

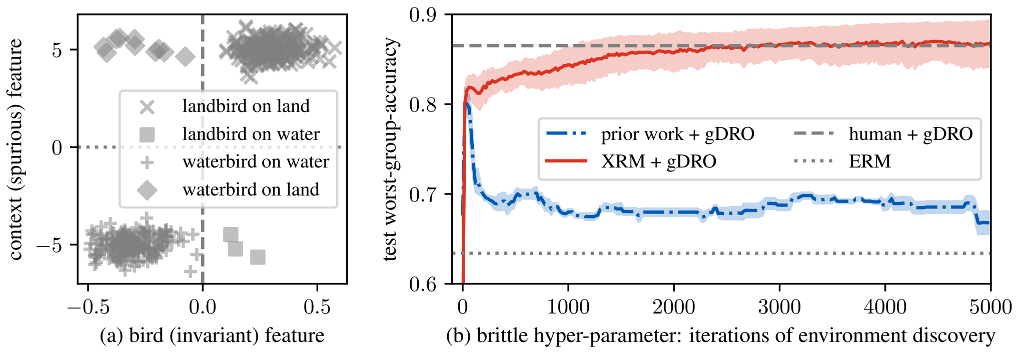


Figure 1: (a) Waterbirds problem with four groups: a *majority group* of waterbirds in water, landbirds in land, waterbirds in land, and a *minority group* of landbirds in water. Models often rely on spurious features to classify the majority of examples and then memorize the minority examples. (b) Worst-group-accuracy (minority) for different methods. (Dotted line) ERM achieves 61%. (Dashed line) GroupDRO with human group annotations (oracle) achieves 87%. (Dashdot blue line) Prior work to discover groups requires early-stopping with surgical precision. (Solid red line) XRM enables an oracle performance of 87% without requiring early stopping.

trained on top of the discovered environments. However, this pipeline suffers from one fundamental issue: the need for precisely controlling the classifier's capacity, ensuring that the discovered environments differ only in spurious correlations. As Figure 1 shows, peak performance (79%) is reached with exact early-stopping in phase-1; without it, accuracy drops to 68%. Lacking a precise indicator of early-stopping, environment discovery methods depend on validation sets with human annotations. Alas, at least in our view, this defeats the *raison d'être* of environment discovery.

Contribution We propose CROSS-RISK MINIMIZATION (XRM), a simple method for environment discovery that requires no human environment annotations whatsoever. XRM trains two twin networks, each holding-in one random half of the training data. During training, XRM instructs each twin to imitate confident held-out mistakes made by their sibling. This results in an "echo-chamber" where twins increasingly rely on bias, converging on a pair of environments that differ in spurious correlation, and share the invariances that fuel downstream out-of-distribution generalization. After twin training, a simple cross-mistake formula allows XRM to annotate all of the training and validation examples with environments. As our experiments show, XRM endows OOD generalization algorithms with oracle-like performance across benchmarks. Returning one final time to Figure 1, we observe that XRM+GroupDRO converges to 87% worst-group-accuracy on Waterbirds, matching the oracle!

The sequel is as follows. Section 2 details the problem formulation. Section 3 surveys prior research on environment discovery. Section 4 describes XRM. Section 5 demonstrates the effectiveness of XRM, while Section 6 discusses its potential shortcomings. Section 7 concludes with thoughts for future work.

2 Learning Invariances Across Environments

The goal of OOD generalization is to build learning systems that perform well beyond the training data distribution. To this end, we collect examples from multiple environments and OOD algorithms search for invariant patterns across these environments, while disregarding environment-specific spurious correlations (Arjovsky et al., 2019). Formally, we seek a predictor f that classifies inputs x into labels y, across all relevant environments $e \in \mathcal{E}$:

$$f \in \operatorname*{argmin}_{\tilde{f}} \sup_{e \in \mathcal{E}} R^e(\tilde{f}), \tag{1}$$

where the risk $R^e(f) = \mathbb{E}_{(x,y)\sim P^e}[\ell(f(x), y)]$ measures the average loss ℓ incurred by the predictor f across examples from environment e, all of them drawn iid from P^e.

In its full generality, OOD generalization in (1) is an admittedly daunting task. To alleviate the burden, prior literature often considers the simplified and more practical version of *sub-population shift* (Sagawa et al., 2019). Given a dataset $\mathcal{D} = \{(x_i, y_i, e_i)\}_{i=1}^n$, the supremum in (1) is replaced by a maximum over the training environments and the risk for each environment is approximated by the empirical risk (Vapnik, 1998). The effectiveness of an OOD algorithm is then assessed by its worst-group-accuracy on a validation set.

In practice, several OOD algorithms have been successful in learning invariances across environments (Gulrajani & Lopez-Paz, 2020; Zhou et al., 2022a; Wang et al., 2021; Yang et al., 2023). Despite their promise, their large-scale application is hindered by the need for human-annotated environments, which are resource-intensive and might be even sub-optimal. Different machine learning models fall prey to different kinds of spurious correlations. In addition, there exists complex interactions between environment definition, function class, distributional shift, and cultural viewpoints (Lopez-Paz et al., 2022). Therefore, environment annotations are helpful only when revealing spurious and invariant patterns under the lens of the learning system under consideration. Could it be possible to design algorithms for the automatic discovery of environments tailored to the learning machine and data at hand?

2.1 Discovering Environments

Nature does not shuffle data—Bottou (2019)

Let us reconsider the problem of OOD generalization without access to environment annotations. This time, it suffices to talk about one training distribution P^{tr} and one testing distribution P^{te}. Our training data is a collection of input-label pairs (x_i, y_i), each drawn iid from the training distribution. While P^{tr} may be the mixture of multiple environments describing interesting invariant and spurious correlations, this rich heterogeneity is shuffled together and unbeknown to us. But, if we could "unshuffle" the training distribution and recover the environments therein, we could invoke the OOD generalization machinery from the previous section and hope for a robust predictor. This is the purpose of automatic environment discovery.

3 Related Work on Environment Discovery

To discover environments, prior work often train a classifier and then assign each training example to two environments based on their loss or classification accuracy. Crucially, one must control the capacity of the classifier with surgical precision, such that it relies only on the spurious correlations. It is only in such cases that the subsequent OOD generalization algorithms can successfully disregard these spurious features.

As a result, proposals for environment discovery differ mainly in how to control the capacity of the classifier. For example, the too-good-to-be-true prior (Dagaev et al., 2021) employs a classifier with a small parameter count while correct-n-contrast (Zhang et al., 2022b, CnC) applies strong weight decay regularization. Just train twice (Zheran Liu et al., 2021, JTT) and environment inference for invariant learning (Creager et al., 2020, EIIL) train a classifier for a limited number of epochs. Learning from failure (Nam et al., 2020, LfF) biases the classifier towards the use of "simple" features by applying a generalized version of the cross entropy loss. Other proposals, such as learning to split (Bao & Barzilay, 2022, LS) and adversarial re-weighted learning (Lahoti et al., 2020, ARL) complement capacity control with adversarial games.

However, all these methods assume having access to a human-annotated validation set to conduct such precise capacity controls. This defeats the purpose of environment discovery. In fact, if we have access to a small dataset with human-annotated environments, these examples suffice to fine-tune the last layer of a deep network towards state-of-the-art worst-group-accuracy (Izmailov et al., 2022).

For a more detailed discussion and related work, please refer to Appendix A.

4 Cross-Risk Minimization (XRM)

We propose CROSS-RISK MINIMIZATION (XRM), an algorithm to discover environments without the need of human supervision. XRM comes with batteries included, namely a recipe for hyper-parameter tuning and a formula to annotate all training and validation data. As we will show in Section 5, environments discovered by XRM endow OOD generalization algorithms with oracle performance.

The blueprint for XRM is as follows. XRM trains two twin classifiers, each holding-in one random half of the training data (Section 4.1). During training, XRM biases each twin to absorb spurious correlation by imitating confident held-out mistakes from their sibling (Section 4.2). XRM chooses hyper-parameters for the twins based on the number of imitated mistakes (Section 4.3). Finally, and given the selected twins, XRM employs a simple "cross-mistake" formula to discover environment annotations for all of the training and validation examples (Section 4.4). Algorithm 1 serves as a companion to the descriptions below; Appendix C contains a PyTorch implementation. The runtime of XRM is akin to one ERM baseline on the training data.

4.1 Twin Setup, Holding-out of Data

We start by initializing two twin classifiers f^a and f^b. Without loss of generality, let these classifiers return softmax probability vectors over the n_{classes} classes in the training data. We split our training dataset $\{(x_i, y_i)\}_{i=1}^{n}$ in two random halves. Formally, we construct a pair of training assignment vectors with entries $m_i^a \sim \text{Bernoulli}(\frac{1}{2})$ and $m_i^b = 1 - m_i^a$, for all $i = 1, \ldots, n$. For classifier f^a, examples with $m_i^a = 1$ are "held-in" and examples with $m_i^a = 0$ are "held-out"; similarly for f^b. Therefore, we will train classifier f^a on training examples where $m_i^a = 1$, and similarly for classifier f^b. See Appendix C for implementation details.

By virtue of this arrangement, we may now estimate the generalization difficulty of any example by looking at the prediction of the twin that held-out such point. This contrasts prior methods, which consume the entire training data, and may therefore conflate generalization and memorization. Here, however, if a point is misclassified when held-out, we see this as evidence of such example belonging

Algorithm 1 Cross-Risk Minimization (XRM)

```
Input: training examples $\{(x_i, y_i)\}_{i=1}^n$ and validation examples $\{(\tilde{x}_i, \tilde{y}_i)\}_{i=1}^m$
Output: discovered environments $\{e_i\}_{i=1}^n$ and $\{\tilde{e}_i\}_{i=1}^m$
• Fix held-in training example assignments $m_i^a \sim \text{Bernoulli}(\frac{1}{2})$ and $m_i^b = 1 - m_i^a$
• Initialize two classifiers $f^a$ and $f^b$
• Until convergence:
    – Compute held-in softmax predictions:
      $p_i^{\text{in}} = m_i^a f^a(x_i) + m_i^b f^b(x_i)$
    – Compute held-out softmax predictions:
      $p_i^{\text{out}} = m_i^b f^a(x_i) + m_i^a f^b(x_i)$
    – Update $f^a$ and $f^b$ to minimize the held-in class-balanced cross-entropy loss $\ell(p^{\text{in}}, y)$
    – Flip $y_i$ into $y_i^{\text{out}} = \text{argmax}_j p_{i,j}^{\text{out}}$, with probability:
      $(p_{i,y_i^{\text{out}}}^{\text{out}} - 1/n_{\text{classes}}) \cdot n_{\text{classes}}/(n_{\text{classes}} - 1)$
• Define cross-mistake function $e(x, y) = [\![(y \notin \text{argmax}_j f^a(x)_j) \vee (y \notin \text{argmax}_j f^b(x)_j)]\!]$
• Discover training $e_i = e(x_i, y_i)$ and validation $\tilde{e}_i = e(\tilde{x}_i, \tilde{y}_i)$ environments
```

to the minority group. Feldman & Zhang (2020) proposes a similar "error when holding-out" construction as a measure of memorization. In the context of label-noise robustness, CrossSplit (Kim et al., 2023) also employs a similar approach, in which, confident held-out mistakes are indicators of a model's memorization of a noisy label.

4.2 Twin Training, Flipping Labels

As Figure 1 shows, the test worst-group-accuracy of an ERM baseline on Waterbirds is 62%. This suggests that, if using ERM to train our twins, each would be able to correctly classify roughly one half of the minority examples. If using these machines to discover environments based on prediction errors, we would dilute the spurious correlation evenly across the two discovered environments. Consequently, it would be difficult for an OOD generalization algorithm to tell apart between invariant and spurious patterns. Albeit counter-intuitive, we would like to hinder the learning process of our twins, such that they increasingly rely on spurious correlation. In the best possible case, the twins would correctly classify all majority examples and misclassify all minority examples, resulting in *zero* worst-group accuracy.

To this end, we propose to steer away our twins from becoming empirical risk minimizers as follows. Let $p_i^{\text{out}} = m_i^b f^a(x_i) + m_i^a f^b(x_i)$ be the held-out softmax prediction for example (x_i, y_i). Also, let $y_i^{\text{out}} = \arg\max_j p_{i,j}^{\text{out}}$ be the held-out predicted class label. Then, at each iteration during the training of the twins, flip y_i into y_i^{out}, with probability,

$$(p_{y_i^{\text{out}}}^{\text{out}} - 1/n_{\text{classes}}) \cdot n_{\text{classes}}/(n_{\text{classes}} - 1), \tag{2}$$

and let each network to minimize their held-in cross-entropy loss—according to the moving targets.

The overarching intuition is that the label flipping Equation (2) implements an "echo chamber" reinforcing the twins to rely on spurious correlation. Label flipping happens more often for confident held-out mistakes and early in training. These are two footprints of spurious correlations, since these are often easier and faster to capture. (In the context of neural networks, this is often referred to as a "simplicity bias" (Shah et al., 2020b; Pezeshki et al., 2021).) Overall, the purpose of Equation (2) is to transform the labels of the training data such that they no longer represent the original classes, but spurious bias. Finally, the adjustment of Equation (2) in terms of n_{classes} ensures low flip probabilities at initialization, where mistakes are random, and not due to spurious correlation. We note that the "echo chamber" effect aligns the twin networks and that is crucially different from methods that use multiple networks to either disagree with or diversify spurious features (Nam et al., 2020; Cha et al., 2021; Rame et al., 2022; Wortsman et al., 2022; Lee et al., 2023; Pagliardini et al., 2023; Lin et al., 2023; Eastwood et al., 2023).

4.3 Twin Model Selection, Counting Label Flips

Before discovering environments, we must commit to a pair of twin classifiers. Each of the twin networks own hyper-parameters, XRM would be incomplete without a model selection criterion (Gulrajani & Lopez-Paz, 2020). We propose to select the twin hyper-parameters showing a maximum number of label flips at the last iteration, and across the training data. To reiterate, by "counting flips" we simply compare the vector of current labels with the vector of original labels—therefore, we do not accumulate counts of double or multiple flips per label. To understand why, recall that each label flip signifies one example that is confidently misclassified when held-out. Therefore, each label flip is evidence about reliance on spurious correlation, which consequently brings us closer to a clear-cut identification of the minority group.

4.4 The Cross-Mistake Formula

Having committed to a pair of twins, we are ready to discover environments for all of our training and validation examples. In particular, we use a simple "cross-mistake" formula to annotate any example (x, y) with the binary annotation, $e(x, y) =$

$$[\![(y \notin \text{argmax}_j f^a(x)_j) \vee (y \notin \text{argmax}_j f^b(x)_j)]\!], \tag{3}$$

Table 1: Worst-group-accuracies, averaged from ten runs across datasets and algorithms, show XRM achieving oracle-level performance. When group labels are not available, class labels substitute them. Additionally, while ERM does not use group labels, it can still benefit from validation group labels for hyperparameter tuning, resulting in improved performance.

	ERM			**GroupDRO**			**RWG**			**SUBG**		
	None	Human	**XRM**	None	Human	**XRM**	None	Human	**XRM**	None	Human	**XRM**
Waterbirds	70.4	76.1	75.3	71.7	88.0	86.1	74.8	87.0	84.5	73.0	86.7	76.3
CelebA	63.3	56.8	57.0	67.8	88.7	88.5	62.8	84.8	81.9	70.6	83.2	82.2
MultiNLI	70.7	72.2	66.7	68.9	75.3	72.8	68.4	71.1	67.2	70.0	67.8	71.8
CivilComments	66.7	73.6	71.9	65.5	73.7	70.1	66.7	74.0	72.4	66.4	71.3	65.6
ColorMNIST	10.1	10.0	13.0	10.0	10.2	69.5	10.1	10.6	70.5	10.1	10.3	64.3
MetaShift	73.8	75.1	74.0	75.4	81.8	78.8	64.6	75.7	78.6	64.6	74.5	77.9
ImagenetBG	78.0	78.7	79.2	78.4	77.6	75.5	76.9	79.3	77.8	79.8	78.8	77.6
Average	61.9	63.2	62.4	62.5	70.8	77.3	60.6	68.9	76.1	62.1	67.5	73.7

where "$\vee$" denotes logical-OR, and "$[\![\,]\!]$" is the Iverson bracket. If operating within the group-shift paradigm, we define one group per combination of label and discovered environment. Notably, the ability to annotate both training and validation examples is a feature inherited from holding-out data during twin training. More particularly, every example—within training and validation sets—is held-out for at least one of the two twins, as subsumed in Equation (3) by the logical-OR operation.

We are now ready to train the OOD generalization algorithm of our choice on top of the training data with environments discovered with XRM.

5 Experiments

This section presents a series of experiments to showcase the effectiveness of XRM on two well-known benchmarks. Additional experiments are also conducted to identify scenarios where XRM excels, as well as scenarios where it fails to discover relevant environments.

5.1 Sub-population Shift Benchmarks

For sub-population shift tasks, we experiment with seven datasets and four algorithms detailed in Appendix B. We compare results with 3 sources of environment annotations:

- None: no env. annotations—class labels are used instead,
- Human: original human-annotated environments,
- XRM: inferred environments discovered by our method.

Metrics Regardless of how training and validation environments are discovered, we always report test worst-group-accuracy over the human environment annotations provided by each dataset. The tables hereby presented show averages over ten random seeds. For results with error bars, see Table 5.

XRM vs. human annotations Table 1 shows that XRM enables oracle-like worst-group-accuracy across datasets. The performance gains are remarkable in the challenging ColorMNIST dataset, where XRM perfectly identifies digits appearing in minority colors, discovering a pair of environments conducive of stronger generalization than the ones originally proposed by humans. For the commonly-reported quartet of Waterbirds, CelebA, MultiNLI, and CivilComments, the average worst-group-accuracy is 67.3% when no group annotations are used. When using XRM, the average worst-group-accuracy significantly improves to 80.4%, closely matching 80.6% achieved with human annotations.

XRM vs. other methods Table 2 compares the worst-group-accuracy achieved by GroupDRO using XRM-inferred environments against other environment discovery methods. These include learning from failure (Nam et al., 2020, LfF), environment inference for invariant learning (Creager et al., 2020, EIIL), just train twice (Zheran Liu et al., 2021, JTT), correct-n-contrast (Zhang et al., 2022b, CnC), automatic feature re-weighting (Qiu et al., 2023, AFR), and LS (Bao & Barzilay, 2022). As seen in the previous subsection, XRM achieves 80.4%, nearly matching oracle performance. The second best method with no access to environment information, JTT, drops to 58.9%. The best method accessing a validation set with human environment annotations, AFR, lags far from XRM, with 78.6%. The computational burden to complete the results from LS was prohibitive, with more details provided in Appendix B.9. For example, one run of LS for Waterbirds, the smallest dataset, took 20 hours. An XRM run for this same dataset, on the same 32GB Volta GPU, takes 10 minutes.

5.2 The DomaninBed Benchmark

Table 3 presents additional domain generalization results on the DomainBed benchmark (Gulrajani & Lopez-Paz,

Table 2: Average/worst accuracies comparing methods for environment discovery. We specify access to annotations in training data (e^{tr}) and validation data (e^{va}). Symbol † denotes original numbers.

			Waterbirds		CelebA		MNLI		CivilComms		Average	
e^{tr}	e^{va}		Avg	Worst	Avg	Worst	Avg	Worst	Avg	Worst	Avg	Worst
✓	✓	ERM	86.1	76.1	93.5	56.8	82.1	72.2	84.6	73.6	86.6	69.7
		GroupDRO	92.6	88.0	93.0	88.7	80.0	75.3	84.2	73.7	87.5	81.4
✗	✓	ERM†	97.3	72.6	95.6	47.2	82.4	67.9	83.1	69.5	89.6	64.3
		LfF†	91.2	78.0	85.1	77.2	80.8	70.2	68.2	50.3	81.3	68.9
		EIIL†	96.9	78.7	89.5	77.8	79.4	70.0	90.5	67.0	89.1	73.4
		JTT†	93.3	86.7	88.0	81.1	78.6	72.6	83.3	64.3	85.8	76.2
		CnC†	90.9	88.5	89.9	88.8	—	—	—	—	—	—
		AFR†	94.4	90.4	91.3	82.0	81.4	73.4	89.8	68.7	89.2	78.6
✗	✗	ERM	85.3	70.4	95.0	63.3	82.3	70.7	81.6	66.7	86.0	67.8
		LfF†	86.6	75.0	81.1	53.0	71.4	57.3	69.1	42.2	77.1	56.9
		EIIL†	90.8	64.5	95.7	41.7	80.3	64.7	—	—	—	—
		JTT†	88.9	71.2	95.9	48.3	81.4	65.1	79.0	51.0	86.3	58.9
		LS†	91.2	86.1	87.2	83.3	78.7	72.1	—	—	—	—
		BAM†	91.4	89.1	88.4	80.1	80.3	70.8	88.3	79.3	87.1	79.8
		XRM	90.6	86.1	91.0	88.5	76.6	72.8	83.5	70.1	85.4	79.4

2020). Experiments compare three settings: ERM without any environment annotations, the CORAL domain generalization algorithm (Sun & Saenko, 2016) with human-annotated environments, and CORAL with environments discovered by XRM. As a note, CORAL is the best performing single-model (non-ensembling) method in the DomainBed suite. Once again, results suggest that the performance when using XRM-inferred annotations is comparable to that of human-annotated environments. Further details on these experiments are provided in Appendix B.2 with full table of results in Table 6.

5.3 Further Analytical Experiments

Label flipping dynamics on Waterbirds Figure 2 explores some of the behaviors of XRM on the Waterbirds dataset. In particular, the top-left panel justifies the use of "percentage of label-flips at convergence" as a model selection criterion for XRM, as it correlates strongly with downstream worst-group-accuracy. The two bottom panels showcase the clear separation of the minority group "landbirds/water" by XRM, as no landbirds in land are in the cross-mistake area. The top-right panel shows that label flipping happens almost exclusively for minority groups, and converges alongside XRM training. This provides XRM with a degree of stability, removing the need for intricate early-stopping criteria.

Revealing spurious correlations in CIFAR-10 with XRM Figure 3 applies XRM to the CIFAR-10 dataset (Krizhevsky et al., 2009). While CIFAR-10 does not contain environment annotations, the discovered environments by XRM for the "plane" and "deer" classes reveal one interesting spurious correlation, namely the background color.

As a final remark, we ablated the need for (i) holding-out data, and (ii) performing label flipping, finding that both components are essential to the performance of XRM.

6 When Does XRM Fail?

In the previous section, we showcased the effectiveness of XRM in successfully discovering relevant environments. However, it is important to note that XRM, like other environment discovery methods, relies on certain assumptions. In the absence of these assumptions, these methods, including XRM, may fail to discover relevant environments. In its full generality, the problem of learning invariant predictors in the absence of appropriate environment annotations is indeed impossible (Lin et al., 2022a). To see that, we note that dividing data into invariance-affording environments parallels the problem of discovering the right causal structure in the field of causal inference (Pearl, 2009). To reveal the true causal structure between an invariant feature X_{inv} and a target Y, one must "control-for" a set of variables/environments E that satisfy the following *conditional independence statement*,

$$Y \perp E \mid X_{\text{inv}}. \tag{4}$$

Unfortunately, different causal structures can produce identical observational data (Peters et al., 2017). Because of that, identifying an appropriate E is impossible without admitting extra knowledge about the causal structure behind our data, and XRM is not an exception of such free lunch.

Table 3: The average and worst test environment accuracies for five datasets in the DomainBed benchmark (Gulrajani & Lopez-Paz, 2020). Three methods are compared: 1) ERM with no environment annotations, 2) CORAL with human-annotated environments, and 3) CORAL with XRM-inferred environments. The model selection is done according to the average accuracy over validation environments.

	VLCS		PACS		OfficeHome		TerraInc		DomainNet	
Method (annotations)	Avg	Worst	Avg	Worst	Avg	Worst	Avg	Worst	Avg	Worst
ERM (None)	77.97	64.85	83.35	72.55	65.47	52.25	47.02	34.60	31.69	9.30
CORAL (Human)	77.87	65.00	84.99	77.70	67.74	53.55	48.51	37.15	41.97	13.25
CORAL (XRM)	77.66	66.15	83.81	77.30	67.01	53.90	49.60	38.00	35.87	11.60

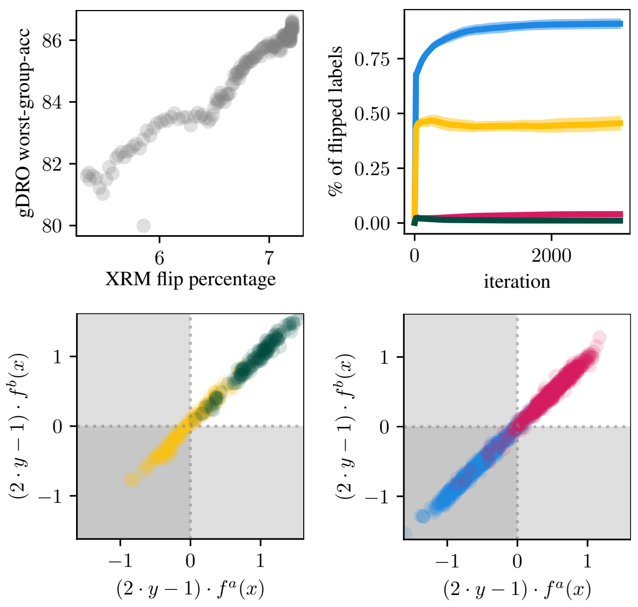


Figure 2: XRM on the Waterbirds problem, concerning 🟥 waterbirds in water, 🟦 waterbirds in land, 🟨 landbirds in water, 🟩 landbirds in land. The top-left panel shows that "percentage of XRM label-flips at convergence" is a strong indicator of "worst-group-accuracy of OOD generalization algorithm", making flips a good criterion to select twin hyper-parameters. The two bottom panels show the signed margin of the twins on each ground-truth group. Each of the bottom plots correspond to one of the classes. Note that a positive margin means correct classification. From each of these class-dependent plots, XRM discovers two environments: one for points in the "mistake-free" white area, and one for points in the "cross-mistake" gray areas. Notably, XRM is able to allocate the two smallest groups 🟦🟨 to dedicated environments. Another notable observation is that the two bottom plots appear as straight lines, indicating that the twin networks agree on their predictions. The top-right panel shows that label flipping happens almost exclusively for the two smallest groups, and stabilizes as training progresses.

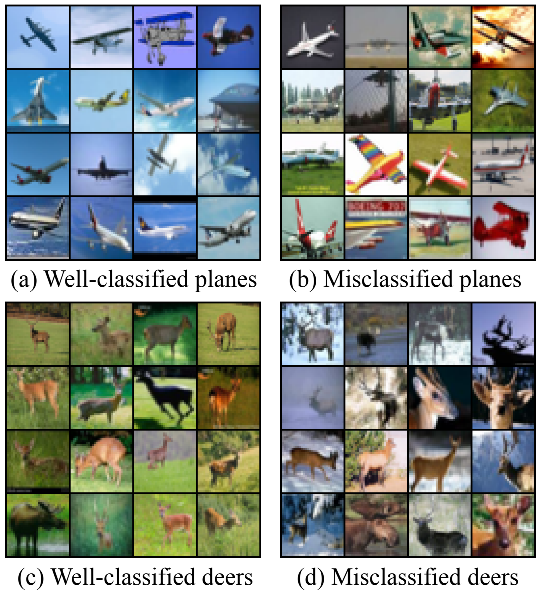


Figure 3: Randomly selected images from CIFAR-10, identified by XRM. Although CIFAR-10 lacks predefined environment annotations, our method has successfully uncovered intriguing environments. Notably, well-classified examples (when held-out) are prototypical, featuring planes in blue skies and deer on green landscapes. In contrast, misclassified examples (when held-out) are less typical, which means they are correctly classified only when included in the training set.

Therefore, we would expect XRM to work well in instances where the inferred environments E satisfy (4), and we should anticipate trouble in those cases where the discovered environments violate (4). On the other hand, as discussed in (Lin et al., 2022a), evaluating (4) requires knowing the invariant feature X_{inv}, which is the variable subject to discovery. This makes (4) difficult to verify in practice when inferring environments E, and the best we can do is to offer some canonical examples of successes and failures, that can guide our choices of when to apply XRM.

Here, we exemplify with four different versions of the ColorMNIST dataset (Arjovsky et al., 2019). All four versions

	CMNIST (Arjovsky et al., 2019)	InvCMNIST (Zhang et al., 2022a)	InveCOLOR (No citation)	MCOLOR (Lin et al., 2022b)
training ($e = 1$)	$S \overset{0.75}{\longrightarrow} Y \overset{0.80}{\longrightarrow} C$	$S \overset{0.85}{\longrightarrow} Y \overset{0.70}{\longrightarrow} C$	$S \overset{0.80}{\longleftarrow} Y \overset{0.75}{\longleftarrow} C$	$S \overset{0.80}{\longleftarrow} Y \overset{0.85}{\longleftarrow} C$
training ($e = 2$)	$S \overset{0.75}{\longrightarrow} Y \overset{0.90}{\longrightarrow} C$	$S \overset{0.85}{\longrightarrow} Y \overset{0.80}{\longrightarrow} C$	$S \overset{0.90}{\longleftarrow} Y \overset{0.75}{\longleftarrow} C$	$S \overset{0.70}{\longleftarrow} Y \overset{0.85}{\longleftarrow} C$
training (pooled)	$S \overset{0.75}{\longrightarrow} Y \overset{0.85}{\longrightarrow} C$	$S \overset{0.85}{\longrightarrow} Y \overset{0.75}{\longrightarrow} C$	$S \overset{0.85}{\longleftarrow} Y \overset{0.75}{\longleftarrow} C$	$S \overset{0.75}{\longleftarrow} Y \overset{0.85}{\longleftarrow} C$
testing	$S \overset{0.75}{\longrightarrow} Y \overset{0.10}{\longrightarrow} C$	$S \overset{0.85}{\longrightarrow} Y \overset{0.10}{\longrightarrow} C$	$S \overset{0.10}{\longleftarrow} Y \overset{0.75}{\longleftarrow} C$	$S \overset{0.10}{\longleftarrow} Y \overset{0.85}{\longleftarrow} C$
inv.feat.	complex, weak	complex, strong	simple, weak	simple, strong
ERM	0.37 ± 0.10	0.67 ± 0.02	$\mathbf{0.75} \pm 0.01$	$\mathbf{0.85} \pm 0.01$
XRM	$\mathbf{0.71} \pm 0.02$	$\mathbf{0.82} \pm 0.02$	0.40 ± 0.01	0.57 ± 0.01
Oracle	0.75	0.85	0.75	0.85

Table 4: Four ColoredMNIST versions, where the environment E influences digit shape S and color C, forming our input $X = (S, C)$. We depict the causal structure for each dataset version, and the correlation between variables. The invariant feature may be the complex digit shape (CMNIST versions) *or* the simple digit color (MCOLOR versions), which in turn could bear the strongest *or* weakest correlation to the target variable—producing four versions of the ColoredMNIST problem. Note that CMNIST - MCOLOR and InverseCMNIST - InverseMCOLOR are indistinguishable from pooled training data alone. At the bottom, test accuracies of ERM, XRM+GroupDRO, and an Oracle which relies solely on the invariant feature.

instantiate a colored digit classification task, differing on whether the invariant feature is "digit shape" or "digit color", and which one of these two variables bear the strongest correlation to the target label. Overall, we expect "digit color" to be faster (easier) to learn, leading to generalization issues when "digit shape"—more difficult and slower to learn—is the desired invariant feature.

We show in Table 4 the average-test-accuracy of ERM and XRM followed by GroupDRO for the four versions of the ColoredMNIST dataset. We also show what a hypothetical oracle, relying solely on the invariant feature, would achieve. ERM performs well when the invariant feature is the simplest of the two. XRM performs well when the invariant feature is the most complex of the two. We highlight that the datasets CMNIST and MCOLOR are observationally equivalent from pooled data alone—and a similar remark follows for InverseCMNIST and InverseMCOLOR. This echoes the impossibility results of (Lin et al., 2022a), namely learning invariant predictors in the absence of environment annotations is impossible in its full generality: for instance, based on training data alone, we would never know if we are dealing with InverseCMNIST or InverseMCOLOR, and therefore we are at a loss of whether to apply ERM or XRM. Nevertheless, XRM remains an state-of-the-art solution for those problems were we would like our learning machine to ignore the fastest-to-learn feature, often being a spurious shortcut (Geirhos et al., 2020; Shah et al., 2020a; Pezeshki et al., 2021), in order to focus on more complex patterns with a higher potential for invariance.

7 Discussion

We have introduced CROSS-RISK MINIMIZATION (XRM), a simple algorithm for environment discovery. XRM provides a recipe to tune its hyper-parameters, does not require early-stopping, and can discover environments for all training and validation data—dropping the requirement for human annotations at all. More specifically, XRM trains two twin classifiers on random halves of the training data, while encouraging each twin to imitate confident held-out mistakes by their sibling. This implements an "echo-chamber" that identifies environments that differ only in spurious correlation, and endow subsequent OOD generalization algorithms with oracle-like performance.

We highlight two directions for future work. Firstly, how does XRM relate to the invariance principle $Y \perp E \mid \Phi(X)$? What is the interplay between revealing relevant labels Y and relevant environments E as to afford invariance? To our knowledge, XRM is the first environment discovery algorithm tampering with labels Y, thus exploring invariance—and the violation thereof—from a new angle. Because relabeling happens with a probability proportional to confidence, we expect model calibration to play a role in understanding the theoretical underpinnings of XRM, as it happened with other invariance methods (Wald et al., 2021). Overall, the theoretical analysis of XRM will call for new tools, because label-flipping steers XRM away from the Bayes-optimal predictor.

Secondly, we would like to further understand the relationship between XRM and the multifarious phenomenon of memorization. Good memorization affords invariance (*Where did I park my car?*), and therefore depends on the collection of environments deemed relevant. Bad memorization happens due to "structured over-fitting", commonly incarnated as a bad learning strategy; "use a simple feature for the majority, then memorize the minority". XRM seems to attack a similar problem but, how does it specifically relate to these two flavours of memorization? Does XRM discover environments that promote features that benefit all examples?

Acknowledgements

We thank Kartik Ahuja, Martín Arjovsky, Léon Bottou, Kamalika Chaudhuri, Cijo Jose, Reyhane Askari Hemmat and Polina Kirichenko for fruitful discussions. We thank the FAIR leadership for their help in publishing this manuscript.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Angwin, J., Larson, J., Mattu, S., and Kirchner, L. Machine bias. *ProPublica*, 2016. URL https://www.propublica.org/article/machine-bias-risk-assessments\-in-criminal-sentencing.

Arjovsky, M., Bottou, L., Gulrajani, I., and Lopez-Paz, D. Invariant risk minimization. *arXiv*, 2019. URL https://arxiv.org/abs/1907.02893.

Bao, Y. and Barzilay, R. Learning to split for automatic bias detection. *arXiv*, 2022. URL https://arxiv.org/abs/2204.13749.

Barocas, S., Hardt, M., and Narayanan, A. *Fairness and Machine Learning: Limitations and Opportunities*. fairmlbook.org, 2019. http://www.fairmlbook.org.

Bell, S. J. and Sagun, L. Simplicity bias leads to amplified performance disparities. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, pp. 355–369, 2023.

Blanchard, G., Lee, G., and Scott, C. Generalizing from several related classification tasks to a new unlabeled sample. In *NeurIPS*, 2011. URL https://proceedings.neurips.cc/paper_files/paper/2011/file/b571ecea16a9824023ee1af16897a582-Paper.pdf.

Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., et al. End to end learning for self-driving cars. *arXiv*, 2016. URL https://arxiv.org/abs/1604.07316.

Borkan, D., Dixon, L., Sorensen, J., Thain, N., and Vasserman, L. Nuanced metrics for measuring unintended bias with real data for text classification. *WWW*, 2019. URL https://arxiv.org/abs/1903.04561.

Bottou, L. Learning representation with causal invariance. *ICLR Keynote*, 2019. URL https://leon.bottou.org/talks/invariances.

Cha, J., Chun, S., Lee, K., Cho, H.-C., Park, S., Lee, Y., and Park, S. SWAD: Domain generalization by seeking flat minima. *NeurIPS*, 2021. URL https://arxiv.org/abs/2102.08604.

Chen, Y., Zhou, K., Bian, Y., Xie, B., Wu, B., Zhang, Y., Ma, K., Yang, H., Zhao, P., Han, B., et al. Pareto invariant risk minimization: Towards mitigating the optimization dilemma in out-of-distribution generalization. *arXiv preprint arXiv:2206.07766*, 2022.

Chen, Y., Bian, Y., Zhou, K., Xie, B., Han, B., and Cheng, J. Rethinking invariant graph representation learning without environment partitions. *ICLR DG Workshop*, 2023. URL https://openreview.net/forum?id=bjw5jqGtDy.

Chen, Y., Bian, Y., Zhou, K., Xie, B., Han, B., and Cheng, J. Does invariant graph learning via environment augmentation learn invariance? *Advances in Neural Information Processing Systems*, 36, 2024.

Creager, E., Jacobsen, J.-H., and Zemel, R. Environment inference for invariant learning. *arXiv*, 2020. URL https://arxiv.org/abs/2010.07249.

Dagaev, N., Roads, B. D., Luo, X., Barry, D. N., Patil, K. R., and Love, B. C. A too-good-to-be-true prior to reduce shortcut reliance. *arXiv*, 2021. URL https://arxiv.org/abs/2102.06406.

DeGrave, A. J., Janizek, J. D., and Lee, S.-I. AI for radiographic COVID-19 detection selects shortcuts over signal. *Nature Machine Intelligence*, 2021. URL https://www.nature.com/articles/s42256-021-00338-7.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Eastwood, C., Singh, S., Nicolicioiu, A. L., Vlastelica, M., von Kügelgen, J., and Schölkopf, B. Spuriosity didn't kill the classifier: Using invariant predictions to harness spurious features. *arXiv*, 2023. URL https://arxiv.org/abs/2307.09933.

Feldman, V. and Zhang, C. What neural networks memorize and why: Discovering the long tail via influence estimation. *arXiv*, 2020. URL https://arxiv.org/abs/2008.03703.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *JMLR*, 2016. URL https://arxiv.org/abs/1505.07818.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2020. URL https://arxiv.org/abs/2004.07780.

Giannone, G., Havrylov, S., Massiah, J., Yilmaz, E., and Jiao, Y. Just mix once: Worst-group generalization by group interpolation. *NeurIPS Workshop on Distribution Shifts*, 2022. URL https://arxiv.org/abs/2210.12195.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv*, 2014. URL https://arxiv.org/abs/1412.6572.

Gulrajani, I. and Lopez-Paz, D. In search of lost domain generalization. *arXiv*, 2020. URL https://arxiv.org/abs/2007.01434.

Hand, D. J. and Henley, W. E. Statistical classification methods in consumer credit scoring: a review. *Journal of the royal statistical society: series a (statistics in society)*, 1997. URL https://www.jstor.org/stable/2983268.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Heaven, W. D. Hundreds of AI tools have been built to catch COVID. none of them helped. *MIT Technology Review*, 2021. URL https://www.technologyreview.com/2021/07/30/1030329/machine-learning-ai-failed//-covid-hospital-diagnosis-pandemic/.

Idrissi, B. Y., Arjovsky, M., Pezeshki, M., and Lopez-Paz, D. Simple data balancing achieves competitive worst-group-accuracy. *CLeaR*, 2022. URL https://arxiv.org/abs/2110.14503.

Izmailov, P., Kirichenko, P., Gruver, N., and Wilson, A. G. On feature learning in the presence of spurious correlations. *NeurIPS*, 2022. URL https://arxiv.org/abs/2210.11369.

Japkowicz, N. The class imbalance problem: Significance and strategies. In *ICML*, 2000. URL https://www.researchgate.net/publication/2639031_The_Class_Imbalance_Problem_Significance_and_Strategies.

Jiang, F., Jiang, Y., Zhi, H., Dong, Y., Li, H., Ma, S., Wang, Y., Dong, Q., Shen, H., and Wang, Y. Artificial intelligence in healthcare: past, present and future. *Stroke and vascular neurology*, 2017. URL https://pubmed.ncbi.nlm.nih.gov/29507784/.

Kim, J., Baratin, A., Zhang, Y., and Lacoste-Julien, S. Crosssplit: mitigating label noise memorization through data splitting. In *International Conference on Machine Learning*, pp. 16377–16392. PMLR, 2023.

Kirichenko, P., Izmailov, P., and Wilson, A. G. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv preprint arXiv:2204.02937*, 2022.

Krizhevsky, A., Nair, V., and Hinton, G. Cifar-10, 2009. URL http://www.cs.toronto.edu/~kriz/cifar.html.

Krueger, D., Caballero, E., Jacobsen, J.-H., Zhang, A., Binas, J., Zhang, D., Le Priol, R., and Courville, A. Out-of-distribution generalization via risk extrapolation (rex). In *ICML*, 2021. URL https://arxiv.org/abs/2003.00688.

LaBonte, T., Muthukumar, V., and Kumar, A. Towards last-layer retraining for group robustness with fewer annotations. *arXiv preprint arXiv:2309.08534*, 2023.

Lahoti, P., Beutel, A., Chen, J., Lee, K., Prost, F., Thain, N., Wang, X., and Chi, E. H. Fairness without demographics through adversarially reweighted learning. *arXiv*, 2020. URL https://arxiv.org/abs/2006.13114.

Lee, Y., Yao, H., and Finn, C. Diversify and disambiguate: Learning from underspecified data. *ICLR*, 2023. URL https://arxiv.org/abs/2202.03418.

Li, G., Liu, J., and Hu, W. Bias amplification enhances minority group performance. *arXiv*, 2023. URL https://arxiv.org/abs/2309.06717.

Liang, W. and Zou, J. Metashift: A dataset of datasets for evaluating contextual distribution shifts and training conflicts. *arXiv*, 2022. URL https://arxiv.org/abs/2202.06523.

Lin, Y., Zhu, S., Tan, L., and Cui, P. Zin: When and how to learn invariance without environment partition? *NeurIPS*, 2022a. URL https://arxiv.org/abs/2203.05818.

Lin, Y., Zhu, S., Tan, L., and Cui, P. Zin: When and how to learn invariance without environment partition? *Advances in Neural Information Processing Systems*, 35: 24529–24542, 2022b.

Lin, Y., Tan, L., Hao, Y., Wong, H., Dong, H., Zhang, W., Yang, Y., and Zhang, T. Spurious feature diversification improves out-of-distribution generalization. *arXiv preprint arXiv:2309.17230*, 2023. URL https://arxiv.org/abs/2309.17230.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *ICCV*, 2015. URL https://arxiv.org/abs/1411.7766.

Lopez-Paz, D., Bouchacourt, D., Sagun, L., and Usunier, N. Measuring and signing fairness as performance under multiple stakeholder distributions. *arXiv*, 2022. URL https://arxiv.org/abs/2207.09960.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Ma, J., Yue, Z., Tomoyuki, K., Tomoki, S., Jayashree, K., Pranata, S., and Zhang, H. Invariant feature regularization for fair face recognition. In *ICCV*, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/papers/Ma_Invariant_Feature_Regularization_for_Fair_Face_Recognition_ICCV_2023_paper.pdf.

Muandet, K., Balduzzi, D., and Schölkopf, B. Domain generalization via invariant feature representation. In *ICML*, 2013. URL https://proceedings.mlr.press/v28/muandet13.html.

Nam, J., Cha, H., Ahn, S., Lee, J., and Shin, J. Learning from failure: Training debiased classifier from biased classifier. *arXiv*, 2020. URL https://arxiv.org/abs/2007.02561.

Pagliardini, M., Jaggi, M., Fleuret, F., and Karimireddy, S. P. Agree to disagree: Diversity through disagreement for better transferability. *ICLR*, 2023. URL https://arxiv.org/abs/2202.04414.

Pang Wei Ko et al. WILDS: A benchmark of in-the-wild distribution shifts. *arXiv*, 2021. URL https://arxiv.org/abs/2012.07421.

Pearl, J. *Causality*. Cambridge university press, 2009. URL http://bayes.cs.ucla.edu/BOOK-2K/.

Peters, J., Bühlmann, P., and Meinshausen, N. Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 2016. URL https://arxiv.org/abs/1501.01332.

Peters, J., Janzing, D., and Schölkopf, B. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017. URL https://mitpress.mit.edu/9780262037310/elements-of-causal-inference/.

Pezeshki, M., Kaba, O., Bengio, Y., Courville, A. C., Precup, D., and Lajoie, G. Gradient starvation: A learning proclivity in neural networks. *Advances in Neural Information Processing Systems*, 34:1256–1272, 2021.

Qiu, S., Potapczynski, A., Izmailov, P., and Wilson, A. G. Simple and fast group robustness by automatic feature reweighting. *arXiv*, 2023. URL https://arxiv.org/abs/2306.11074.

Rame, A., Kirchmeyer, M., Rahier, T., Rakotomamonjy, A., Gallinari, P., and Cord, M. Diverse weight averaging for out-of-distribution generalization. *NeurIPS*, 2022. URL https://arxiv.org/abs/2205.09739.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *ICLR*, 2019. URL https://arxiv.org/abs/1911.08731.

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

Shah, H., Tamuly, K., Raghunathan, A., Jain, P., and Netrapalli, P. The pitfalls of simplicity bias in neural networks. *NeurIPS*, 2020b. URL https://arxiv.org/abs/2006.07710.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Sohoni, N., Dunnmon, J., Angus, G., Gu, A., and Ré, C. No subclass left behind: Fine-grained robustness in coarse-grained classification problems. *NeurIPS*, 2020. URL https://arxiv.org/abs/2011.12945.

Sun, B. and Saenko, K. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV Workshops*, 2016.

Tan, X., Yong, L., Zhu, S., Qu, C., Qiu, X., Yinghui, X., Cui, P., and Qi, Y. Provably invariant learning without domain information. In *Proceedings of the 40th International Conference on Machine Learning*, 2023. URL https://proceedings.mlr.press/v202/tan23b.html.

Teney, D., Abbasnejad, E., and van den Hengel, A. Unshuffling data for improved generalization in visual question answering. In *ICCV*, 2021. URL https://arxiv.org/abs/2002.11894.

Tsirigotis, C., Monteiro, J., Rodriguez, P., Vazquez, D., and Courville, A. Group robust classification without any group information. *arXiv preprint arXiv:2310.18555*, 2023.

Vapnik, V. *Statistical learning theory*. Wiley, 1998. URL https://www.wiley.com/en-us/Statistical+Learning+Theory-p-9780471030034.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. The Caltech-UCSD birds-200-2011 dataset. *California Institute of Technology*, 2011. URL https://www.vision.caltech.edu/datasets/cub_200_2011/.

Wald, Y., Feder, A., Greenfeld, D., and Shalit, U. On calibration and out-of-domain generalization. *NeurIPS*, 2021. URL https://arxiv.org/abs/2102.10395.

Wang, J., Lan, C., Liu, C., Ouyang, Y., Qin, T., Lu, W., Chen, Y., Zeng, W., and Yu, P. S. Generalizing to unseen domains: A survey on domain generalization. *arXiv*, 2021. URL https://arxiv.org/abs/2103.03097.

Williams, A., Nangia, N., and Bowman, S. R. A broad-coverage challenge corpus for sentence understanding through inference. *ACL*, 2017. URL https://aclanthology.org/N18-1101/.

Woodward, J. *Making things happen: A theory of causal explanation*. Oxford university press, 2005.

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *ICML*, 2022. URL https://arxiv.org/abs/2203.05482.

Xiao, K., Engstrom, L., Ilyas, A., and Madry, A. Noise or signal: The role of image backgrounds in object recognition. *arXiv*, 2020. URL https://arxiv.org/abs/2006.09994.

Yang, Y., Zhang, H., Katabi, D., and Ghassemi, M. Change is hard: A closer look at subpopulation shift. *arXiv*, 2023. URL https://arxiv.org/pdf/2302.12254.pdf.

Yao, H., Wang, Y., Li, S., Zhang, L., Liang, W., Zou, J., and Finn, C. Improving out-of-distribution robustness via selective augmentation. *arXiv*, 2022. URL https://arxiv.org/abs/2201.00299.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. URL https://arxiv.org/abs/1710.09412.

Zhang, J., Lopez-Paz, D., and Bottou, L. Rich feature construction for the optimization-generalization dilemma. In *International Conference on Machine Learning*, pp. 26397–26411. PMLR, 2022a.

Zhang, L., Li, J.-F., and Wang, W. Semi-supervised domain generalization with known and unknown classes. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Zhang, M., Marklund, H., Dhawan, N., Gupta, A., Levine, S., and Finn, C. Adaptive risk minimization: Learning to adapt to domain shift. *NeurIPS*, 2021. URL https://arxiv.org/abs/2007.02931.

Zhang, M., Sohoni, N. S., Zhang, H. R., Finn, C., and Ré, C. Correct-n-contrast: A contrastive approach for improving robustness to spurious correlations. *arXiv preprint arXiv:2203.01517*, 2022b.

Zheran Liu, E., Haghgoo, B., Chen, A. S., Raghunathan, A., Koh, P. W., Sagawa, S., Liang, P., and Finn, C. Just train twice: Improving group robustness without training group information. *arXiv*, 2021. URL https://arxiv.org/abs/2107.09044.

Zhou, K., Liu, Z., Qiao, Y., Xiang, T., and Loy, C. C. Domain generalization: A survey. *IEEE PAMI*, 2022a. URL https://arxiv.org/abs/2103.02503.

Zhou, X., Lin, Y., Zhang, W., and Zhang, T. Sparse invariant risk minimization. In *Proceedings of the 39th International Conference on Machine Learning*, 2022b. URL https://proceedings.mlr.press/v162/zhou22e.html.

A Further Related Work

The literature on OOD and domain generalization spans a decade and comprises a vast amount of works. In the review below, we survey (i) some of the major milestones of OOD generalization research, (ii) advances in the sub-problem of sub-population shift, (iii) the multifarious connections between OOD generalization and causal inference, (iv) efforts to learn domains, sub-populations, or environments from pooled collections of training examples previous to our XRM proposal, and (v) their limitations in terms of annotation requirements and impossibility results.

(i) The first works in *OOD and domain generalization* proposed algorithms that summarize each domain as a kernel mean embedding of the respective distribution of inputs (Blanchard et al., 2011; Muandet et al., 2013); these were later extended to the realm of deep neural networks (Zhang et al., 2021). One common avenue towards domain generalization is to learn a predictor where the feature representation has the same distribution across domains (Sun & Saenko, 2016; Ganin et al., 2016). Another major strategy is to enforce learning a richer feature space (Zhang et al., 2022a), which can be done by combining the weights of multiple models with different hyper-parameter configurations (Cha et al., 2021; Rame et al., 2022; Wortsman et al., 2022; Lin et al., 2023), or biasing training to make them disagree with each other (Nam et al., 2020; Pagliardini et al., 2023; Lee et al., 2023). Learning from combinations of examples, by means of mixup (Zhang et al., 2018), is also a promising route to diminish the impact of spurious correlations (Yao et al., 2022; Giannone et al., 2022). All in all, there are multiple frameworks that evaluate dozens of OOD generalization algorithms across a variety of benchmark datasets, such as DomainBed (Gulrajani & Lopez-Paz, 2020) and WILDS (Pang Wei Ko et al., 2021). We recommend the reader to consult recent surveys (Zhou et al., 2022a; Wang et al., 2021) for a taxonomy of the vast array algorithms on offer.

(ii) *Sub-population shift* is a particular type of OOD generalization problem, where environments are direct annotations of a spurious attribute, and one can assume that the test domain will be equal a subdistribution—group—of the training data. The gold-standard for addressing sub-population shifts is group distributionally robust optimization (Sagawa et al., 2019, GroupDRO). Group subsampling and reweighting schemes, albeit simple, also provide state-of-the-art accuracy (Idrissi et al., 2022). To achieve good performance, it is known that it suffices to finetune the last layer of a deep neural network with a small training set with balanced groups (Izmailov et al., 2022). The framework of SubpopBench compares twenty algorithms for sub-population shift across a dozen benchmark datasets (Yang et al., 2023).

(iii) The goal of OOD/domain generalization can be understood as finding predictors invariant across a family of relevant environments (Arjovsky et al., 2019). This establishes an intimate link between OOD generalization and causality under the interventionist account, where causation is defined as invariance across interventions (Woodward, 2005). A pioneering method attacks the problem OOD generalization as finding invariant causal predictors (Peters et al., 2016, ICP). The framework of invariant risk minimization (IRM) (Arjovsky et al., 2019) extends ICP to deep neural networks, advocating the invariance principle of "finding a feature representation such that the optimal classifier matches across environments". Researchers have proposed multiple variants of the original IRM formulation, with notable examples being risk extrapolation (Krueger et al., 2021, vREX) and sparse risk minimization (Zhou et al., 2022b). The IRM framework has found multiple applications, with fair face recognition (Ma et al., 2023) being a recent example.

(iv) The main factor limiting the application of OOD generalization and sub-population shift machinery is their requirement of domain, environment, or group annotations. Unfortunately, these are resource-intensive to obtain and are limited by human annotators' biases, as the biases they identify may not align with those learned by models, and vice versa (Bell & Sagun, 2023).

Consequently, a wide array of methods has been recently proposed to estimate these annotations from pooled collections of training data (LaBonte et al., 2023; Zhang et al., 2023; Tsirigotis et al., 2023). Among them, learning from failure (Nam et al., 2020, LfF) learns a biased network, and a final network that focuses on the examples misclassified by the biased network. Environment inference for invariant learning (Creager et al., 2020, EIIL) searches for an environmental partition that violates the IRM principle. Just-train-twice (Zheran Liu et al., 2021, JTT) trains one first network for a few iterations, and a final network to focus more on the examples from the first network. Correct and Contrast (Zhang et al., 2022b, CNC) leverages ERM failures and contrastive learning to learn a robust representation. Automatic feature reweighting (Qiu et al., 2023, AFR) learns a first network for a few iterations, and then fine-tunes the last layer to focus on mistakes. Learning to split (Bao & Barzilay, 2022, LS) and adversarial re-weighted learning (Lahoti et al., 2020, ARL) implement adversarial games to find a split of the training data inducing maximum out-of-sample error. Bias amplification (Li et al., 2023, BAM) incorporates per-example "slack variables" to absorb the fast learning of spurious correlations. No subclass left behind (Sohoni et al., 2020, GEORGE) clusters the hidden representation of a neural network to construct different environments. (Teney et al., 2021) manually identify variables to stratify pooled collections of training examples into environments. In the context of label noise robustness, the prior CrossSplit (Kim et al., 2023) uses a similar approach to that of XRM. CrossSplit uses a pair of networks trained on two disjoint parts of the training data and uses label correction to prevent memorization of noisy (mis-labelled) examples. While confidently mistaken cross-predictions are indicators of a model's memorization of a noisy label in the context of the CrossSplit paper, they indicate a model's reliance on spurious correlations in our work. In the context of graph OOD generalization, GALA (Chen et al., 2024) was proposed as a framework for graph OOD generalization that uses an assistant ERM model to identify invariant subgraphs through agreement and disagreement with proxy predictions.

(v) One important note to the environment discovery methods described above is that they still require group annotations in a validation set, used for selecting a model with good worst-group-accuracy. In the complete absence of environment annotations, learning invariant predictors is an impossible task in its full generality (Chen et al., 2022; Lin et al., 2022a; Tan et al., 2023; Chen et al., 2023). Because we have proposed XRM as an alternative to surmount such daunting task, the next section provides intuitions to identify success and failure cases of our method.

B Experimental Details

B.1 Sub-population Shift Experiments

For the results in Table 1, we follow standard experimental protocol as in (Yang et al., 2023). For model selection, we adhere to the standard practice of using the worst-group-accuracy. We try 10 different hyper-parameter combinations detailed in Appendix B.7 with one random seed. We select the hyper-parameter combination and early-stopping iteration yielding maximal validation worst-group-accuracy (or, in the absence of groups, worst-class-accuracy). Next, we repeat the experiment 10 times with different seeds. Regardless of how training and validation groups are discovered, we always report test worst-group-accuracy over the **human** group annotations provided by each dataset.

We consider six standard datasets. These are the four image datasets Waterbirds (Wah et al., 2011), CelebA (Liu et al., 2015), MetaShift (Liang & Zou, 2022), and ImageNetBG (Xiao et al., 2020); and the two natural language datasets MultiNLI (Williams et al., 2017) and CivilComments (Borkan et al., 2019). For CelebA, predictors map pixel intensities into a binary "blonde/not-blonde" label. No individual face characteristics, landmarks, keypoints, facial mapping, metadata, or any other information was used to train our CelebA predictors. Image datasets use a pretrained ResNet-50 (He et al., 2016), frozen for XRM experiments. Text datasets use a pretrained BERT (Devlin et al., 2018). The linear layers on the top of the pretrained models are initialized at zero. All images are resized and center-cropped to 224×224 pixels, and undergo no data augmentation. We use SGD with momentum 0.9 to learn from image datasets unless otherwise mentioned, and we employ AdamW (Loshchilov & Hutter, 2017) with default $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for text benchmarks. As for the OOD generalization algorithms, we consider ERM, group distributionally robust optimization (Sagawa et al., 2019, GroupDRO), group re-weighting (Japkowicz, 2000, RWG), and group sub-sampling (Idrissi et al., 2022, SUBG).

We also conduct experiments on ColorMNIST (Arjovsky et al., 2019), but keep a strict protocol. More specifically, we set *both* training and validation data to contain two environments, with 0.8 and 0.9 label-color correlation, while the test environment shows 0.1 label-color correlation. This contrasts Arjovsky et al. (2019), who used the test environment for model selection. We train a three-layer fully-connected network with layer sizes $[2 * 14 * 14, 300, 300, 2]$ and use ReLU as the activation function. The network is optimized using the Adam optimizer with a learning rate of $1e-3$, and default parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We train all algorithms for a number of iterations that allows convergence within a reasonable compute budget. Full results with error bars are reported in Table 5.

For the experiment on CIFAR-10 (Krizhevsky et al., 2009), we train a VGG-16 model (Simonyan & Zisserman, 2014) using SGD with a learning rate of $1e-2$ and a momentum of 0.9.

B.2 DomainBed Experiments

For the results in Table 3 and Table 6, we adhere to the original codebase from DomainBed (Gulrajani & Lopez-Paz, 2020). Each dataset is used to train a model on all but one environment, which is held out as the test domain. This process is repeated for each possible environment as the test domain, resulting in multiple training and testing splits for each dataset. We experiment with ERM and the CORAL algorithm as it is the best performing single-model (non-ensembling) method

Table 5: Worst-group-accuracies and standard deviations for all datasets, algorithms, and annotations over 10 trials.

	ERM			**GroupDRO**			**RWG**			**SUBG**		
	None	Human	**XRM**	None	Human	**XRM**	None	Human	**XRM**	None	Human	**XRM**
Waterbirds	70.4 ±2.99	76.1 ±2.37	75.3 ±1.96	71.7 ±4.09	88.0 ±2.61	86.1 ±1.28	74.8 ±2.50	87.0 ±1.63	84.5 ±1.53	73.0 ±2.75	86.7 ±1.00	76.3 ±8.41
CelebA	63.3 ±2.73	56.8 ±3.48	57.0 ±3.48	67.8 ±1.29	88.7 ±1.67	88.5 ±1.23	62.8 ±1.32	84.8 ±1.45	81.9 ±2.56	70.6 ±2.13	83.2 ±2.70	82.2 ±2.54
MultiNLI	70.7 ±4.04	72.2 ±3.02	66.7 ±3.41	68.9 ±2.65	75.3 ±1.42	72.8 ±1.77	68.4 ±1.77	71.1 ±1.60	67.2 ±1.56	70.0 ±2.97	67.8 ±0.66	71.8 ±1.58
CivilComments	66.7 ±3.46	73.6 ±5.77	71.9 ±4.48	65.5 ±2.05	73.7 ±0.60	70.1 ±4.41	66.7 ±4.58	74.0 ±6.30	72.4 ±0.93	66.4 ±2.56	71.3 ±7.63	65.6 ±1.12
ColorMNIST	10.1 ±0.51	10.0 ±2.40	13.0 ±2.27	10.0 ±0.51	10.2 ±2.37	69.5 ±0.98	10.1 ±0.51	10.6 ±1.85	70.5 ±1.00	10.1 ±0.51	10.3 ±2.21	64.3 ±1.09
MetaShift	73.8 ±2.99	75.1 ±4.62	74.0 ±3.95	75.4 ±3.91	81.8 ±4.42	78.8 ±4.81	64.6 ±4.55	75.7 ±4.45	78.6 ±4.86	64.6 ±3.38	74.5 ±3.49	77.9 ±5.58
ImagenetBG	78.0 ±3.04	78.7 ±1.89	79.2 ±1.93	78.4 ±3.43	77.6 ±1.40	75.5 ±1.69	76.9 ±2.42	79.3 ±2.65	77.8 ±2.66	79.8 ±3.55	78.8 ±3.55	77.6 ±1.79

according to the DomainBed suite. For each triplet of (dataset, method, test environment), we sweep over 10 different hyper-parameter combinations detailed in Appendix B.7. We perform model selection based on the *average accuracy over the validation environments*, which is referred to as the *'training domain validation set'* in the DomainBed paper. In Table 6, we report the results for each environment when selected as the test environment. Additionally, we report the average and worst environment test accuracies. In settings without environment annotations (i.e., ERM), we combine all training environments and then split into one training and one validation set. In those cases with annotations, whether human-annotated or discovered by XRM, each training environment is divided into as many training and validation sets as the number of environments.

Table 6: Full results for the DomainBed suite.

VLCS	C	L	S	V	Avg	Worst
ERM (None)	96.70	64.85	74.20	76.15	77.97	64.85
CORAL (Human)	97.35	65.00	72.80	76.35	77.87	65.00
CORAL (XRM)	95.55	66.15	72.45	76.50	77.66	66.15

PACS	A	C	P	S	Avg	Worst
ERM (None)	84.65	80.65	95.55	72.55	83.35	72.55
CORAL (Human)	84.90	80.75	96.60	77.70	84.98	77.70
CORAL (XRM)	81.90	77.30	96.90	79.15	83.81	77.30

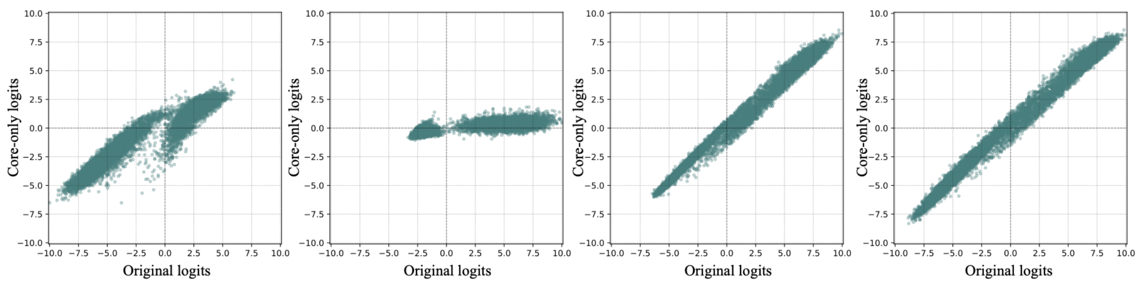
OfficeHome	A	C	P	R	Avg	Worst
ERM (None)	59.50	52.25	74.15	76.00	65.47	52.25
CORAL (Human)	64.00	53.55	76.15	77.25	67.73	53.55
CORAL (XRM)	61.80	53.90	74.85	77.50	67.01	53.90

TerraIncognita	L100	L38	L43	L46	Avg	Worst
ERM (None)	54.80	42.30	56.40	34.60	47.02	34.60
CORAL (Human)	58.20	39.25	59.45	37.15	48.51	37.15
CORAL (XRM)	59.20	45.10	56.10	38.00	49.60	38.00

DomainNet	clip	info	paint	quick	real	sketch	Avg	Worst
ERM (None)	47.40	14.75	37.45	9.30	42.10	39.15	31.69	9.30
CORAL (Human)	60.05	20.25	47.90	13.25	59.95	50.45	41.97	13.25
CORAL (XRM)	50.40	16.80	42.30	11.60	50.40	43.70	35.87	11.60

B.3 Dominoes Experiments

We conduct an additional experiment on the dataset of Dominoes (Geirhos et al., 2020). Dominoes is an image dataset in which images from two other datasets are concatenated. Particularly, the top half of an image shows MNIST digits



(a) ERM (b) XRM (c) XRM+GroupDRO (d) Grnd-truth annot.+GroupDRO

Figure 4: **Logit scatter plots for models trained on Dominoes-MF.** The x-axis shows the logits with the original training examples, while the y-axis displays the logits for the same training examples with the MNIST part removed. These plots help reveal the extent of reliance on the spurious feature (MNIST digit) versus the core feature (FashionMNIST or Cifar). **a)** Vanilla ERM Model: Strong reliance on the spurious MNIST digit feature and partial dependence on the core feature. **b)** XRM Model: Strong reliance on the spurious MNIST digit. **c)** GroupDRO Model with XRM-Inferred annotations: point mostly on the diagonal suggesting invariance to the spurious feature, focusing only on the core feature. **d)** GroupDRO with ground-truth annotations for reference. We highlight that for XRM, it is desirable to only rely on the spurious feature since this will then enable the subsequent GroupDRO to learn the invariant core feature.

from classes $\{0, 1\}$. The bottom half shows images from classes $\{$coat, dress$\}$ of FashionMNIST in the MF (MNIST-MNISTFashion) version. And in the MC (MNIST-CIFAR) version, the bottom half shows images from classes $\{$car, truck$\}$ of CIFAR-10. We followed a setup as described in (Kirichenko et al., 2022) (except that we used a smaller ConvNet) where the spurious MNIST part is strongly correlated (99% of the time) with the labels in the training data but drops to 50% (random) in the validation and test sets. Meanwhile, the core feature (either FashionMNIST or CIFAR) is always fully correlated with the labels. We compare three methods on these datasets: the Vanilla ERM, GroupDRO) using the ground-truth annotations[1], and GroupDRO with XRM-inferred annotations. The results, shown in Table 7, suggest that the group annotations inferred by XRM are as effective as the ground-truth annotations.

Table 7: Worst-group accuracy of three methods on two Dominoes datasets. Dominoes MF and MC are concatenation of MNIST digits with FashionMNIST and Cifar-10, respectively. The MNIST digit is spurious with 99% correlation in the training set and random correlation in the validation and test sets. The core feature (either FashionMNIST or Cifar) is however always fully correlated with the labels. GT denotes ground-truth group annotations.

	ERM	XRM+GroupDRO	GT+GroupDRO
Dominoes MF	50.74	86.68	85.28
Dominoes MC	48.30	68.78	69.43

To further compare these methods, in Figure 4, we provide a scatter plot of the model logits for each method. Specifically, we feed two versions of all training examples to a fully trained model and store two sets of logits. One set is the result of the original training examples, and the other set is obtained with the MNIST part removed (replaced by average).

- For an ERM model, the logits vary mostly along the x-axis and less along the y-axis, suggesting that the model is mostly relying on the spurious feature but also partially on the core feature.
- An XRM model varies mostly along the x-axis and is almost invariant along the y-axis, indicating that the model strongly relies on the spurious feature alone and hence can effectively split the data into two environments according to the spurious correlation.
- For a model trained with GroupDRO on XRM-annotations, points are almost all on the diagonal, suggesting that removing the spurious feature does not change the logits much, implying that the model is invariant to the spurious feature and only relies on the core feature.
- A model trained with GroupDRO using the ground-truth annotations shows similar results, proving it mainly relies on the core feature.

[1]annotations indicate whether the MNIST half matches the label or not

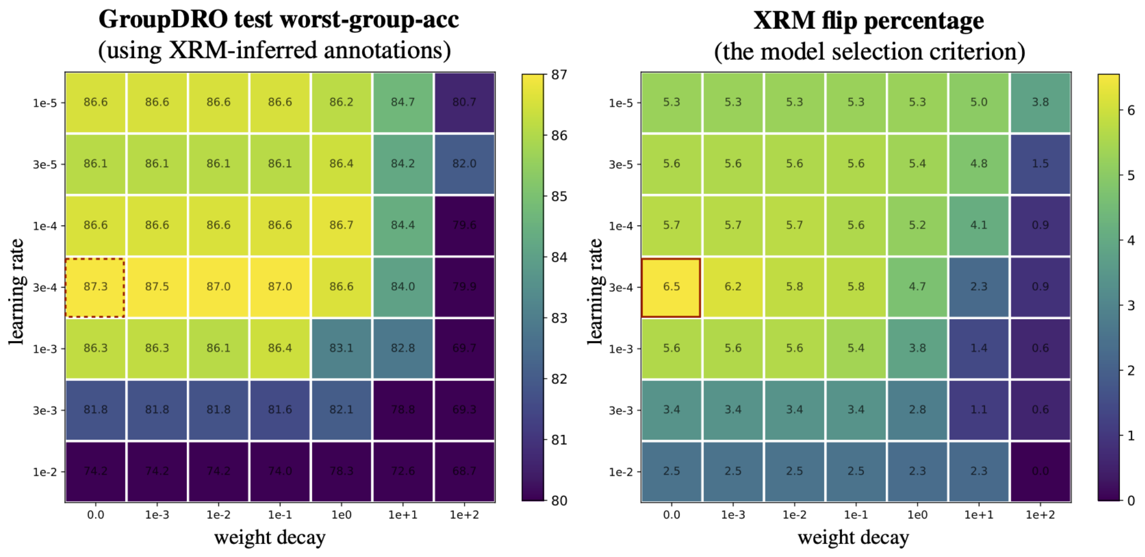


Figure 5: **Hyperparameter sensitivity analysis of XRM on the Waterbirds.** Here we evaluate the sensitivity of XRM's performance to variations in learning rate and weight decay hyperparameters, while keeping the batch size fixed at 512. **Left:** worst-group-acc of a GroupDRO model trained with XRM annotations. Each cell within the grid represents a hyperparameter combination for XRM, with the color intensity indicating the test worst-group-accuracy. **Right:** the percentage of labels flipped by XRM for the corresponding hyperparameter combination, serving as the model selection criterion. **Analysis:** The best XRM model is chosen according to the highest flip percentage (red square), which resulted in an accuracy of 87.3% (dashed red square). However, the plot on the left shows that even neighboring cells with very different hyperparameters can still lead to near-optimal performance.

B.4 Hyperparameter Sensitivity Analysis of XRM on Waterbirds

Here, we conduct a hyperparameter sensitivity experiment on the Waterbirds dataset and report the results in Figure 5. We fix the batch-size at 512 and vary the learning rate and weight decay coefficient to see how XRM performs with different hyperparameters. This grid of outcomes across different learning rate and weight decay settings shows that XRM works well across a wide range of hyperparameter combinations.

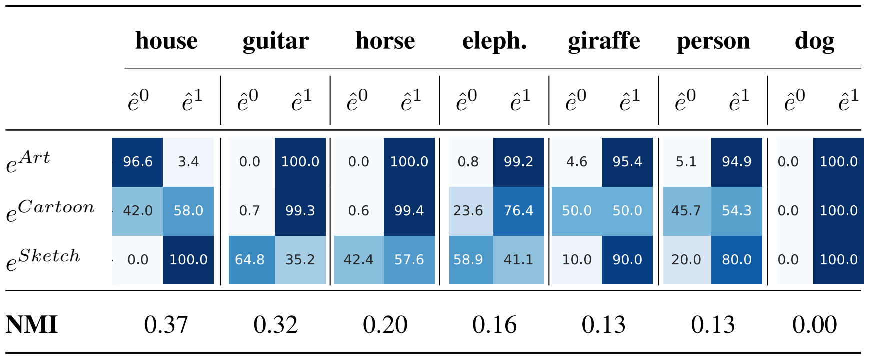
For each combination of learning rate and weight decay (each cell in the grid), we fully train an XRM model, get the group annotations, and then use GroupDRO with these inferred annotations. The hyperparameters for GroupDRO kept the same throughout all experiments. Hence, any variability in performance is only attributed to the XRM phase.

B.5 Can XRM Handle Settings with More Than Two True Underlying Environments?

XRM is always presented with pooled data with no explicit annotations. Regardless of how many human-annotation environments are in the pooled data, XRM always splits the data into 2 environments (2 groups per label). For example, all the datasets in table 6, have more than two human-annotated environments. That is while XRM split the examples into two environments.

Whether two environments suffice—when constructed appropriately—is in fact a fascinating question that we have pondered about for a while. Our experiments, especially those on the DomainBed, show that although XRM splits the data into a smaller number of groups than the human annotations, it is successful in forming sufficiently varied groups to match the performance with human annotations.

To illustrate, take one of the experiments with the PACS dataset where the "Photo" environment is left for testing and the other three environments of "Art", "Cartoon", and "Sketch" are pooled together and presented to XRM. Now XRM's objective is to split the dataset into environments that lead to learning an invariant predictor, not to recover the original environment. XRM's splitting is done according to the 'hardness' of examples and might or might not align with the human annotations. To see that, we visualize a per-class confusion matrix for the environment discovery of XRM. We also quantify



	house		guitar		horse		eleph.		giraffe		person		dog	
	$\hat{e}^0$	$\hat{e}^1$	$\hat{e}^0$	$\hat{e}^1$	$\hat{e}^0$	$\hat{e}^1$	$\hat{e}^0$	$\hat{e}^1$	$\hat{e}^0$	$\hat{e}^1$	$\hat{e}^0$	$\hat{e}^1$	$\hat{e}^0$	$\hat{e}^1$
e^{Art}	96.6	3.4	0.0	100.0	0.0	100.0	0.8	99.2	4.6	95.4	5.1	94.9	0.0	100.0
$e^{Cartoon}$	42.0	58.0	0.7	99.3	0.6	99.4	23.6	76.4	50.0	50.0	45.7	54.3	0.0	100.0
e^{Sketch}	0.0	100.0	64.8	35.2	42.4	57.6	58.9	41.1	10.0	90.0	20.0	80.0	0.0	100.0
NMI	0.37		0.32		0.20		0.16		0.13		0.13		0.00	

Figure 6: **Confusion matrices of XRM-inferred environments vs. human-annotated environments for the PACS dataset.** Each row corresponds to one of the human-annotated environments, $e \in \{\text{Art, Cartoon, Sketch}\}$. Columns represent the two environments, $\hat{e}^{0/1}$ inferred by XRM for each class. Entries in the matrices indicate the percentage of examples from each human-annotated environment that XRM grouped into each of its two inferred environments. Normalized Mutual Information (NMI), a metric between zero and one, quantifies the alignment between the inferred and annotated environments. **Analysis:** For certain classes (e.g., "house"), the XRM-inferred environments align well with human annotations, while for other classes (e.g., "dog"), no alignment is observed. This is acceptable as long as the inferred environments can be used for invariant learning, which appears to be the case. XRM's goal is to find environments suitable for subsequent invariant learning, rather than necessarily recovering the original environments. As a reference point, a random splitter would result in an NMI of zero on all classes, and a subsequent invariant learning method based on random split cannot perform better than ERM. Worst-group-accuracy of ERM is 72.55 while CORAL with Human: 77.70, XRM: 77.30, random envs: 71.60.

the alignment between XRM environments and human-annotated ones using Normalized-Mutual-Information (NMI), a higher NMI means more alignment.

We observe that for some of the classes (e.g., the "House" class) there is a rather strong alignment. However, for some other classes (e.g., the "Dog" class), there is no alignment between human-annotated environments and those of XRM. That is absolutely fine as long as the inferred environments can be used for invariant learning and that appears to be the case.

As a reference point, a random splitter would lead to an NMI of zero on all classes and a subsequent invariant learning method cannot do better than an ERM with no environment annotations.

B.6 XRM Model Selection

To determine the best hyper-parameter combination for each experiment, we run XRM with 10 different hyper-parameter combinations detailed in Appendix B.7 and one random seed. We then compute the percentage of flipped labels that appear at the last iteration for each combination. The combination with the highest percentage is selected as the best. We disregard rare cases in which a hyper-parameter combination yields degenerate grouping where every example in one class in misclassified. Next, we repeat the experiment 10 times with different seeds to obtain 10 sets of inferred labels. Finally, we apply an OOD generalization algorithm on these inferred labels again using 10 different seeds.

B.7 Hyper-parameter Sampling Grids

algorithm	**hyper-parameter**	**ResNet**	**BERT**
XRM, ERM, SUBG, RWG	learning rate	$10^{\text{Uniform}(-5,-3)}$	$10^{\text{Uniform}(-6,-4)}$
	weight decay	$10^{\text{Uniform}(-6,-3)}$	$10^{\text{Uniform}(-6,-3)}$
	batch size	$2^{\text{Uniform}(5,7)}$	$2^{\text{Uniform}(4,6)}$
	dropout	—	$\text{Random}([0, 0.1, 0.5])$
GroupDRO	η	$10^{\text{Uniform}(-3,-1)}$	$10^{\text{Uniform}(-3,-1)}$

B.8 Statistics of the Datasets

Dataset	Data type	Number of envs.	Number of classes	Dataset size
Waterbirds	Image	2	2	11788
CelebA	Image	2	2	202599
CivilComments	Text	8	2	242436
MultiNLI	Text	2	3	412349
MetaShift	Image	2	2	3499
ImageNetBG	Image	2	9	192255
VLCS	Image	4	5	10729
PACS	Image	4	7	9991
OfficeHome	Image	4	65	15588
Terra Incognita	Image	4	10	24788
DomainNet	Image	6	345	586575

B.9 Learning to Split on Waterbirds

We benchmarked the official learning to split code-base `https://github.com/YujiaBao/ls` on the WaterBirds dataset. We found on a Volta-32GB GPU running the learning to split group inference module took approximately 20 hours. We assessed the method's sensitivity to two hyperparameters: the number of epochs used for early stopping (`patience` argument in the codebase) and the pre-supposed ratio of groups (based on the `ratio` argument in the code). For `patience` we swept over (2, 5, 10) with 5 being the default value. For `ratio`, we swept over (0.25, 0.5, 0.75) with 0.75 being the default value based on the paper. We found worst group performance using a fixed GroupDRO phase-2 training varied by as much as $\pm 7\%$ on Waterbirds.

C XRM in PyTorch

```python
import torch
from torch.nn.functional import cross_entropy

def balanced_cross_entropy(p, y):
    losses = torch.nn.functional.cross_entropy(p, y, reduction="none")
    return sum([losses[y == yi].mean() for yi in y.unique()])

def xrm(x_tr, y_tr, x_va, y_va, lr=1e-2, max_iters=1000):
    # init twins and assign examples (Section 4.1)
    nc = len(y_tr.unique())
    net_a = torch.nn.Linear(x_tr.size(1), nc)
    net_b = torch.nn.Linear(x_tr.size(1), nc)
    net_a.weight.data.mul_(0.0)
    net_b.weight.data.mul_(0.0)
    ind_a = torch.zeros(len(x_tr), 1).bernoulli_(0.5).long()

    # training (Section 4.2)
    opt = torch.optim.SGD(
        list(net_a.parameters()) + list(net_b.parameters()), lr)

    for iteration in range(max_iters):
        pred_a, pred_b = net_a(x_tr), net_b(x_tr)
        pred_hi = pred_a * ind_a + pred_b * (1 - ind_a)
        pred_ho = pred_a * (1 - ind_a) + pred_b * ind_a

        opt.zero_grad()
        balanced_cross_entropy(pred_hi, y_tr).backward()
        opt.step()

        # label flipping, useful for model selection (Section 4.3)
        p_ho, y_ho = pred_ho.softmax(dim=1).detach().max(1)
        is_flip = torch.bernoulli((p_ho - 1 / nc) * nc / (nc - 1)).long()
        y_tr = is_flip * y_ho + (1 - is_flip) * y_tr

    # environment discovery (Section 4.4)
    cm = lambda x, y: torch.logical_or(
        net_a(x).argmax(1).ne(y),
        net_b(x).argmax(1).ne(y)).long().detach()

    return cm(x_tr, y_tr), cm(x_va, y_va)
```

The code above may be helpful to clarify our exposition in the main text. For an end-to-end example running linear XRM and GroupDRO, see: `https://github.com/facebookresearch/XRM/quick_run.py`.