Failure Modes of Domain Generalization Algorithms

Tigran Galstyan\textsuperscript{1,3}, Hrayr Harutyunyan\textsuperscript{2}, Hrant Khachatrian\textsuperscript{1,4}, Greg Ver Steeg\textsuperscript{2}, Aram Galstyan\textsuperscript{2}
\textsuperscript{1} YerevaNN, \textsuperscript{2} USC Information Sciences Institute, \textsuperscript{3} Russian-Armenian University, \textsuperscript{4} Yerevan State University

Abstract

Domain generalization algorithms use training data from multiple domains to learn models that generalize well to unseen domains. While recently proposed benchmarks demonstrate that most of the existing algorithms do not outperform simple baselines, the established evaluation methods fail to expose the impact of various factors that contribute to the poor performance. In this paper we propose an evaluation framework for domain generalization algorithms that allows decomposition of the error into components capturing distinct aspects of generalization. Inspired by the prevalence of algorithms based on the idea of domain-invariant representation learning, we extend the evaluation framework to capture various types of failures in achieving invariance. We show that the largest contributor to the generalization error varies across methods, datasets, regularization strengths and even training lengths. We observe two problems associated with the strategy of learning domain-invariant representations. On Colored MNIST, most domain generalization algorithms fail because they reach domain-invariance only on the training domains. On Camelyon-17, domain-invariance degrades the quality of representations on unseen domains. We hypothesize that focusing instead on tuning the classifier on top of a rich representation can be a promising direction.

1. Introduction

Over the past decade machine learning research was predominantly focused on settings where the learner observes training data from an unknown distribution and is evaluated on testing data, sampled from the same distribution. While modern deep learning approaches excel in such settings, they do significantly worse when the test data comes from a different distribution [32, 41]. These methods might rely on dataset biases to perform well, and fail when those biases are eliminated [4, 10].

The goal of generalizing beyond training distribution is formulated in the domain generalization (DG) task, where the learner observes training data from multiple domains and is evaluated on unseen domains. Naturally, it is assumed that training and testing domains have some invariant properties or mechanisms, which allow generalization from one to another. At a high level, all domain generalization approaches seek to capture those invariances, but do that differently. A few of the possible directions of achieving domain generalization are: learning domain-invariant representations [9, 24], learning class-conditioned domain-invariant representations [8, 21, 45], using robust loss functions [36], learning invariant causal predictors [3], and using meta-learning [20]. Most of the methods listed above outperform the straightforward empirical risk minimization (ERM) approach on toy domain generalization instances (e.g., colored MNIST). However, Gulrajani and Lopez-Paz [14] demonstrate that when evaluated on realistic DG instances, these methods are unable to outperform ERM significantly. To improve domain generalization methods or propose new ones, we need to understand why and how do domain generalization methods fail. This is the main goal of this paper.

Our contributions are threefold. First, we characterize the general failure modes of domain generalization methods: training set underfitting, test set inseparability, training-test misalignment and classifier non-invariance. We develop tools that measure the contribution of each of these failures in the total error of a given model. Inspired by the popularity of the methods based on invariant representation learning, we also characterize failure modes related to achieving domain invariance. Second, we identify two common patterns of generalization failures. In the first pattern, many algorithms achieve domain-invariant representations across the training domains, but not on unseen domains, while the generalization error is negatively correlated with the representation invariance on unseen domains. The second pattern is when domain invariance is increased across all domains, but the increase coincides with a degradation of the representations of unseen domains, thus limiting the accuracy of the models. Third, we show that by fixing the representations it is possible to isolate the classifier non-invariance failure, and significantly improve the generalization even with the most basic algorithms. These findings additionally confirm that domain-invariant representations are neither necessary nor sufficient for successful domain generalization.

The code for measuring contributions of failure modes is available at github.com/YerevaNN/dom-
replicating the experiments is available at github.com/TigranGalstyan/wilds.

2. Related Work

The ability to generalize beyond the training distribution is the key goal of machine learning. Torralba and Efros [41] show that common image classification datasets have significant differences, because of which methods trained on one dataset often fail to generalize well to other datasets. The fact that learning on a single domain is susceptible to dataset biases and spurious correlations has been confirmed in many contexts [2, 4, 6, 10, 32, 41].

A few settings focus on generalization outside the training distribution. The out-of-distribution (OOD) and robustness literature focus on the case when there is a distribution shift at test time [30, 39], including but not limited to label shifts [22, 34], co-variate shifts [12, 38], conditional shifts [44], visual distortions [7, 16], stylistic and other changes [15]. More general settings of domain generalization and domain adaptation assume that examples from multiple domains are available for training, with the difference that in domain adaptation a collection of examples (labeled or not) from the testing domain are available for adaptation [43]. In this paper we focus on the domain generalization problem (also called zero-shot domain adaptation [28]), because of its generality and better correspondence with practical settings. Nevertheless, most of the proposed techniques and definitions can be easily extended to domain adaptation.

A group of approaches aim to get domain generalization by learning domain-invariant representations [8, 9, 21, 24, 45]. DANN [9] uses an adversarial classifier to predict domain from representations, while C-DANN [21] learns such a classifier for each domain separately. Galstyan et al. [8] regularize empirical risk minimization (ERM) with a term that uses Hilbert-Schmidt independence criterion (HSIC) [11] to make representations be independent from domains conditioned on labels. Zhao et al. [45] use a variety of techniques to enforce the distribution of labels conditioned on representations be the same for all domains. DeepCORAL [40] adds a regularization term to align the second-order statistics of representations of different domains. Invariant risk minimization (IRM) [3] aims to learn invariant causal predictors, by finding a representation of data such that optimal classifiers on top of representations are the same for each domain. Li et al. [21] use meta-learning with the one-step look-ahead gradient update technique to simulate evaluating on unseen domains during the training. GroupDRO [36] minimizes the worst-case risk across training domains. Recently, methods based on gradient matching have been proposed [31, 37]. Finally, a few works introduce benchmarks for evaluating domain generalization methods [14, 18].

Gulrajani and Lopez-Paz [14] demonstrate that none of the existing domain generalization approaches outperform empirical risk minimization when evaluated on realistic tasks. There is a very limited amount of research done on why and how these domain generalization methods fail. Rosenfeld et al. [33] study the failure modes of the IRM in theoretical settings. Nagarajan et al. [25] explain the mechanisms by which ERM fails on very easy out-of-domain generalization tasks. In contrast to these works, our analysis and proposed techniques below are applicable to any domain generalization algorithm.

3. Problem Setting and Notation

Consider an input space $\mathcal{X}$ and output space $\mathcal{Y}$. A joint probability distribution $p(x, y)$ on $\mathcal{X} \times \mathcal{Y}$ is called a domain and defines a prediction task. In the domain generalization task we assume there is a family $\mathcal{D}$ of domains that are somehow related to each other and correspond to similar prediction tasks. The learner observes training data from $n_1$ domains, $p^1_1(x, y), \ldots, p^1_{n_1}(x, y)$, and $n_3$ test domains, $p^3_1(x, y), \ldots, p^3_{n_3}(x, y)$. We assume that there is no label shift across domains: $p^1_1(y) = \ldots = p^1_{n_1}(y) = p^2(y) = \ldots = p^2_{n_2}(y) = p^3(y) = \ldots = p^3_{n_3}(y)$. We discuss this limitation in Appendix F. We also assume that for each domain $i = 1, \ldots, n_j$, $j = 1, 2, 3$, we are given a collection of independent samples $\mathcal{D}_i^j$ from the corresponding distribution $p_i^j(x, y)$. Each of these sets is further divided into two parts: $\mathcal{D}_i^j = \mathcal{T}_i^j \cup \mathcal{V}_i^j$. For simplicity, we also define the union of the samples of training (validation, testing) domains: $\mathcal{D}^j = \bigcup_{i=1}^{n_j} \mathcal{D}_i^j$, $\mathcal{T}^j = \bigcup_{i=1}^{n_j} \mathcal{T}_i^j$, $\mathcal{V}^j = \bigcup_{i=1}^{n_j} \mathcal{V}_i^j$, for each $j = 1, 2, 3$. The algorithms will be trained on samples from $\mathcal{T}^j$. The set $\mathcal{V}^1$ is called in-domain validation set in [18], quasi-development set in [8] and training-domain validation set in [14]. It is used to measure the performance of the algorithms on unseen samples from the training domains. The performance on unseen domains is measured using $\mathcal{D}^3$. The sets $\mathcal{T}^j$ and $\mathcal{V}^j, j = 2, 3$, are used for analysis.

We consider domain generalization methods that use neural networks with two components: a feature extractor $z = h_\theta(x) \in \mathbb{R}^d$ with parameters $\theta \in \Theta$, and a classification head $\tilde{y} = f_\omega(z) \in \mathbb{R}^C$ with parameters $w \in \mathcal{W}$. Throughout the paper we call $z$ the representation of $x$. Let $\ell : \mathbb{R}^C \times \mathcal{Y} \to \mathbb{R}$ be a loss function that measures discrepancy between a prediction and a label. In this work $\ell$ is chosen to be the standard 0-1 loss function: $\ell(\tilde{y}, y) = 1(\argmax_k \tilde{y}_k(z(x)) \neq y)$, which is the most pop-
4. Failure Modes

We propose simple evaluation metrics to diagnose a trained model and identify a set of failures that contribute to the final error on unseen domains. We present simplified schematic visualizations of 2-dimensional representation spaces corresponding to each of the failure modes, e.g. Fig. 1. In all such figures each circle corresponds to a representation of one sample. The domain of a sample is encoded by the color of its circle. Orange-red-pink colors correspond to the training domains, while green-blue colors correspond to the test domains. Filled and empty circles are used to encode the binary labels.

To formally define the generalization and invariance metrics, we need the following additional notation. Let \((X_1^1, Y_1^1), \ldots, (X_{n_1}^1, Y_{n_1}^1)\) be random variables drawn from training domains \(p_{1}^{1}(x, y), \ldots, p_{n_1}^{1}(x, y)\) and \((X_3^1, Y_3^1), \ldots, (X_{n_3}^3, Y_{n_3}^3)\) be random variables drawn from test domains \(p_{1}^{3}(x, y), \ldots, p_{n_3}^{3}(x, y)\). Let \((X^{1,3}, Y^{1,3})\) be a random variable drawn from the mixture of all training and test domains \(p^{1,3}(x, y) = \frac{1}{n_1 + n_3} (\sum_{i=1}^{n_1} p_{i}^{1}(x, y) + \sum_{i=1}^{n_3} p_{i}^{3}(x, y))\).

4.1. Generalization metrics

Below we define four evaluation metrics that capture qualitatively different aspects of generalization. All of this metrics will be some kind of errors (so lower better). Hence, we will use terms “metric” and “error” interchangeably.

Training set underfitting. How well does the model perform on training domains? Formally, this metric is denoted by \(e'_0\) and is defined as follows:

\[
e'_0 \triangleq \frac{1}{n_1} \sum_{i=1}^{n_1} \mathbb{E}_{X_i^1, Y_i^1} \left[ \ell \left( f_w(h_\theta(X_i^1)), Y_i^1 \right) \right],
\]

where \(h_\theta\) and \(f_w\) are the learned feature extractor and classifier, respectively. This metric is expected to be small for most domain generalization algorithms. A model might have large \(e'_0\) if the regularization terms of its objective were so strong compared to the classifier loss that the feature extractor failed to learn anything useful. An example will be discussed in the Experiments section. Training set underfitting is demonstrated in Fig. 1a.

Test set inseparability. How well are the representations of the test domains separable with respect to the chosen class of classifier heads? Formally, we define test set inseparability error as follows:

\[
e'_1 \triangleq \inf_{w' \in \mathcal{W}} \left\{ \frac{1}{n_3} \sum_{i=1}^{n_3} \mathbb{E}_{X_i^3, Y_i^3} \left[ \ell \left( f_{w'}(h_\theta(X_i^3)), Y_i^3 \right) \right] \right\}.
\]

This metric will be large for models whose feature extractor is overfitted on the training domains and does not produce a reasonable representation for the test domains. Note that the quality of the representation is measured with respect to the class of classifier heads (e.g., linear functions), because we are not interested in cases when the representations have information about domains but that information is not decodable/usable by the considered family of classifiers. The case when \(e'_0 = 0\) and \(e'_1\) is large is demonstrated in Fig. 1b.

Training-test misalignment. Is there a common classifier for the representations of training and test domains? The extent of the answer to this question being negative is measured by the \(e'_2\) metric:
This case is demonstrated in Fig. 1c. Such scenarios can be able to find a classifier that works for both training and test domains with significant success. In such cases, if the learned model on test domains. Formally, the metric is the standard test error and measures the performance across domains.

\[ e_2' = \frac{1}{n_3} \sum_{i=1}^{n_3} \mathbb{E}_{X_i^2, Y_i^2} \left[ \ell(f_{\tilde{w}}(h_{\theta}(X_i^2)), Y_i^2) \right], \]

where \( \tilde{w} \in \arg \min_{w \in \mathcal{W}} \mathbb{E}_{X_1^3, Y_1^3} \left[ \ell(f_w(h_{\theta}(X_1^3)), Y_1^3) \right]. \)

If \( e_1' \) is small, the representations of the samples from each domain are good enough to be separated by a domain-specific classifier. But it might be impossible to find a classifier that would separate samples from all domains (both training and test) at once, resulting in relatively high \( e_2' \) error. This case is demonstrated in Fig. 1c. Such scenarios can be thought of as “milder” versions of overfitting, as the feature extractor learned useful and decodable information, but was not able to distribute the representations in a consistent way across domains.

**Classifier non-invariance.** This final generalization metric is the standard test error and measures the performance of the learned model on test domains. Formally, the metric \( e_3' \) is defined as follows:

\[ e_3' = \frac{1}{n_3} \sum_{i=1}^{n_3} \mathbb{E}_{X_i^3, Y_i^3} \left[ \ell(f_w(h_{\theta}(X_i^3)), Y_i^3) \right]. \]

If \( e_3' = 0 \), the representations are so good that there exists a classifier that can separate samples from both training and test domains with significant success. In such cases, \( e_3' \) essentially measures whether the training algorithm was able to find a classifier that works for both training and test domains. Hence, the name of this metric \( e_3' \) is “classifier non-invariance”. Note that a training algorithm might easily fail to select an invariant classifier, as it does not have access to the test data during training. This scenario is schematically demonstrated in Fig. 1d.

Note that the four failure modes defined above and depicted in Fig. 1 are extreme cases and should not be thought as a comprehensive list of cases. In Sec. 4.4 we propose a way of attributing the model’s overall test error to each of these failure modes.

### 4.2 Invariance metrics

Many domain generalization algorithms attempt to learn domain-invariant representations. The intuition is that if the representations of the samples are similar across all domains, then a classifier designed for the training domains will generalize to the test domains. As the algorithms have access only to the training data, they usually achieve domain invariance across the training domains, but not across the union of training and test domains.

Motivated by the generalization metrics defined above, we introduce metrics that assess qualitatively different aspects of domain invariance of learned representation. While there are many ways to measure the extent of invariance of representations across two or more domains (e.g., using formal distances or divergences between probability distributions), we choose to use a technique similar to the one used in generalization metrics above. Informally, we will measure how invariant are the representations of samples of...
two domains by measuring how well one can differentiate them using a domain classifier. Importantly, we will draw domain classifier from the same family of functions as label classifiers. That is, if the label classifier head uses a specific architecture, we would use the same architecture (with a different number of outputs) for domain classifiers. This intentionally ignores domain information in representations that cannot be decoded by the label classifier. Domain classifiers will be functions of the form \( g_\omega : \mathbb{R}^d \rightarrow \mathbb{R}^K, \omega \in \Omega \), where \( K \) is the number of domains.

**Training domain distinguishability.** Are the representations of examples from the training domains domain-invariant? Formally, we denote our first invariance metric \( d_0' \) and define it the following way:

\[
d_0' \triangleq 1 - \inf_{\omega \in \Omega} \left[ \frac{1}{n_1} \sum_{i=1}^{n_1} \mathbb{E}_{X^1_i} \left[ \ell(g_\omega(h_\theta(X^1_i)), i) \right] \right] - \frac{1}{n_1}.
\]

Here the constant \( \frac{1}{n_1} \) is the accuracy of the random classifier. Note that lower values of \( d_0' \) correspond to higher invariance, and \( d_0' = 0 \) implies it is impossible to distinguish domains better than the trivial baseline. Large values of \( d_0' \) can arise when the regularizer designed to induce invariance is too weak compared to the classification loss. In practice, most algorithms do not end up in this state if the regularization term is tuned correctly. It is important to note that achieving domain invariance on the training domains is not necessary to have domain generalization, as demonstrated in Fig. 1e.

**Training-test domain distinguishability.** Are the representations of examples from the union of the training and test domains domain-invariant? Our second invariance metric \( d_1' \) measures the extent of this question having a positive answer:

\[
d_1' \triangleq 1 - \inf_{\omega \in \Omega} \left[ \frac{1}{n_1 + n_3} \sum_{i=1}^{n_1} \mathbb{E}_{X^1_i} \left[ \ell(g_\omega(h_\theta(X^1_i)), i) \right] \right] - \frac{1}{n_1 + n_3}.
\]

Assuming that \( d_0' = 0' \), this metric \( d_1' \) can be large if the distributions of the representations of the training and test sets do not coincide; if there is no domain invariance across the test domains; or both. In theory, this failure of reaching invariance can coincide with all failure modes of generalization, as shown in Fig. 2. Moreover, it is possible to have domain generalization along with large \( d_1' \) (Fig. 2e).

Even when a model achieves high training-test domain invariance (low \( d_1' \)), it is still possible that representations of Class 1 samples of the first domain coincide with the representations of Class 2 samples of the second domain and vice versa. In this case, there will be no domain generalization. The next metric is designed to capture such situations.

**Training-test class-conditional domain invariance.** Are the representations of samples belonging to each of the classes domain-invariant? Let \( E_y \) denote the event \( \{Y^1_1 = y \land \cdots \land Y^1_{n_1} = y \land Y^3_{1} = y \land \cdots \land Y^3_{n_3} = y \} \). We define the \( d_2' \) metric as follows:

\[
d_2' \triangleq 1 - \frac{1}{C} \sum_{y=1}^{C} \inf_{\omega \in \Omega} \mathbb{E} \left[ \frac{1}{n_1 + n_3} \sum_{i=1}^{n_1} \ell(g_\omega(h_\theta(X^1_i)), i) \right. \\
\left. + \sum_{i=1}^{n_3} \ell(g_\omega(h_\theta(X^3_i)), i + n_1) \right| E_y \right] - \frac{1}{n_1 + n_3}.
\]

This \( d_2' \) metric can be seen as the conditional version of the previous metric \( d_1' \).

4.3. Relations between domain invariance and generalization metrics

We prove two propositions that establish connections between generalization and invariance failures. In particular, these propositions rule out some combinations of invariance and generalization failures. We first formally define domain invariance of representations and class-conditional domain invariance of representations (not to be confused with the corresponding invariance metrics).

**Definition 4.1 (Domain invariance of representations).** Let \( \mathcal{D} \) be a family of domains. We say that the representations learned by a feature extractor \( z = h(x) \) are domain-invariant with respect to \( \mathcal{D} \) if for any two domains \( p_1(x, y) \) and \( p_2(x, y) \) from \( \mathcal{D} \), the distributions of representations are equal, i.e., \( \forall z, p_{Z_1}(z) = p_{Z_2}(z) \), where \( X_1 \sim p_1(x), X_2 \sim p_2(x) \), \( Z_1 = h(X_1) \) and \( Z_2 = h(X_2) \).

Likewise, we define invariance of representations conditioned on labels by requiring \( p(z|y) \) to be the same for across all domains of the family \( \mathcal{D} \).

**Proposition 1.** If \( e_2' = 0 \), then domain invariance of representations w.r.t. the union of training and testing domains implies class-conditional domain invariance w.r.t. the union of training and testing domains.

**Proposition 2.** If \( e_0' = 0 \) and representations are class-conditional domain-invariant w.r.t. the union of training and testing domains, then \( e_3' = 0 \).

The second proposition implies that for a class-conditional domain-invariant model, perfect performance on the training domains is sufficient for domain generalization.

4.4. Decomposition of errors

The generalization metrics defined above are sequential in the sense that if \( e_i' \) is large then \( e_{i+1}' \) is likely to be large as well. For this reason, the differences \( e_{i+1}' - e_i' \), \( i = 0, 1, 2 \), are more suitable quantities for analysis purposes. In fact,
the failure modes depicted in Fig. 1 are cases when one of these differences is large in conjunction with the previous metric being close to zero. Following this reasoning, we define $e_0 \triangleq e'_0$, $e_1 \triangleq e'_1 - e'_0$, $e_2 \triangleq e'_2 - e'_1$, $e_3 \triangleq e'_3 - e'_2$, and decompose the error $e \triangleq e'_3$ on the test domains into four components:

$$
e = e_0 + e_1 + e_2 + e_3.$$

(1)

Each of these components can be interpreted as the individual contribution of the corresponding failure mode to the error of the model. This decomposition is the most meaningful when the its components are nonnegative. In general, $e_1$, $e_2$, and $e_3$ can be negative, for example when samples of testing domains are significantly easier to classify compared to those of training domains. However, this is a rare phenomenon and has not been observed in our experiments. Moreover, one can prove that on average (w.r.t. to the training-test splits of the domains) each $e_i$, $i = 0, 1, 2, 3$ is nonnegative. We give the precise formulation of this statement along with its proof in Appendix B.

Similarly, domain-distinguishability $d \triangleq d'_2$ can be decomposed into three components:

$$
d = d_0 + d_1 + d_2.
$$

(2)

where $d_0 \triangleq d'_0$, $d_1 \triangleq d'_1 - d'_0$, and $d_2 \triangleq d'_2 - d'_1$. Again, each of these components can be interpreted as the individual contribution of the corresponding failure mode. Some components of this decomposition also can be negative in rare situations but are nonnegative if we consider averaging over training-test splits of the domains (see Appendix B).

5. Experiments

5.1. Datasets and algorithms

The number of datasets suitable for testing domain generalization algorithms is not large. An attempt to collect them under a unified format was done in [18]. Two of the seven datasets are based on MNIST digits, four others are simple unions of unrelated datasets with similar labels, and only one is realistic: TerraIncognita. This latter one has a serious label shift between domains. Recently proposed WILDS benchmark [18] has another seven datasets which are connected to real-world problems. Only one of the seven, Camelyon17, is carefully designed to have no label shift. It is a patch-based variant of the larger Camelyon17 dataset [5] of lymph node captures. The version in the WILDS benchmark contains 450000 96x96 patches of cancer metastases in lymph node sections. The label for each patch is binary indicating whether the patch contains any tumor tissue. The domain of a patch is the hospital from where the image comes from. There are five different hospitals, three for training, one for validation and one for testing. Following [18], we use DenseNet-121 [17] for all algorithms on this dataset and train for 10 epochs.

We also use a slightly modified version of Colored MNIST dataset from [8], originally from [19]. We changed the dataset generation process to mimic Camelyon17. Namely, we construct five domains by splitting MNIST images into five sets of equal size. For each domain we randomly fix three “colors”, which are essentially 30 dimensional vectors, one for each digit. Then we “colorize” each image with the corresponding color. We end up with around 1200 images in each domain of shape 50x28x28. All models trained on this dataset use a simple neural network with a two-layer ReLU-activated convolutional feature extractor and a linear layer on top of it. All models are trained for 10 epochs. Samples from the datasets are presented in Appendix C.

We analyze the following domain generalization algorithms. Empirical Risk Minimization (ERM) [42] is used as a baseline. It minimizes the sum of errors on all training domains and does not use domain information. ERM+HSIC [8] is a simple algorithm that attempts to induce domain invariance for training domains by adding a Hilbert-Schmidt Indepence Criterion (HSIC) [12] regularization term that penalizes domain information in the learned representations. DeepCORAL [40] was introduced as a domain adaptation algorithm, but was recently applied in domain generalization settings [14]. It penalizes the difference between the means and covariance matrices of representation distributions across domains. Invariant Risk Minimization (IRM) [3] tries to push representation distributions for all domains to have the same optimal classifier head. In [29] the authors analyze the phenomenon of gradient starvation in algorithms based on gradient descent. They propose a new regularization method called Spectral Decoupling (SD). To overcome gradient starvation, it penalizes model’s confidence by adding an L2 penalty on networks logits. GroupDRO was used by [35] to tackle poor worst-group performance. It tries to increase worst-group performance by avoiding spurious correlations in the training data. By interpreting domains as groups, this algorithm becomes applicable to our setting. Another method relying on domain-invariant features is Domain-Adversarial Neural Network (DANN) introduced in [1]. It tries to achieve invariance by using a domain classifier (from features) and a gradient reversal layer. All of these algorithms (except ERM) have a regularization strength hyperparameter, which we denote by $\beta$. 

\[ e_0 \triangleq e'_0, e_1 \triangleq e'_1 - e'_0, e_2 \triangleq e'_2 - e'_1, e_3 \triangleq e'_3 - e'_2, \]

\[ + e_0 + e_1 + e_2 + e_3. \]
Hyperparameter ranges can be found in Appendix D.

Following literature, we consider the outputs of the last hidden layer as the learned representations \( h_\theta(x) \). Hence, label and domain classifiers \( f_w(z) \) are linear (i.e., just a single fully-connected layer).

**Model selection.** The authors of DomainBed benchmark explicitly warned against improper model selection methods in domain generalization. They advocated that in practice model selection should be done either based on the performance on in-domain validation sets (\( \mathcal{V}_1 \)), or by using leave-one-domain-out validation. WILDS benchmark introduced separate validation domains and suggested to use them for model selection. In our experiments we will use this method for hyperparameter selection. To select the best checkpoint inside a single run, one can use the same validation domains. In this paper we always picked the last epoch.

5.2. Measuring the failures

To avoid accessing the test domains, we perform analysis on training and validation domains only. The metrics defined in Sec. 4 contain expectations over distributions and infimums over classifiers. In practice, we approximate infimums by empirical risk minimization using the implementation of logistic regression in scikit-learn package [27]. Note that this might fail to find the optimum, and the empirical estimates might be worse than the true values. In case of \( \epsilon_3' \), the estimate can be worse than the solution found by the domain generalization algorithm itself, i.e. \( \epsilon_3' < \epsilon_2' \). In fact, we encounter this phenomenon later in Tab. 1. Following standard machine learning practices, we approximate distributions \( p_i^1(x, y) \) and \( p_i^2(x, y) \) with corresponding empirical distributions \( \mathcal{T}_i^1 \) and \( \mathcal{T}_i^2 \) during training, and with \( \mathcal{V}_i^1 \) and \( \mathcal{V}_j^2 \) when reporting the scores (\( i = 1, \ldots, n_1, j = 1, \ldots, n_2 \)). In practice, to make \( d_i^1 \) and \( d_i^2 \) comparable with \( d_i' \), we train and evaluate domain classifiers on the union of \( n_1 - n_2 \) training domains and \( n_2 \) validation domains.

6. Results and Discussion

Our analysis shows distinct patterns of failures on the two datasets. On Colored MNIST, all algorithms, including the ERM baseline (\( \beta = 0 \)), achieve domain invariance on training domains, but unseen domains remain quite distinguishable. At the same time, the representations of each domain remain separable (\( \epsilon_0 \) and \( \epsilon_1 \) are close to zero), but we have well-expressed training-test misalignment and/or classifier non-invariance across all algorithms and hyperpar-
As we have only two unseen domains in Camelyon17, we look at both of them during our analysis (violating “do not look at the test set” rule). Many methods we tried, including the baseline, increase the invariance after the first few epochs of training. The large variance of $e_1$ is observed only after $d_1$ gets small (Figs. 3e and 3f). Instead, when the domains are well distinguishable, $e_1$ is relatively stable (although the absolute values are different on the two domains) and is accompanied by $e_2$ and $e_3$. In fact, on the test domain, the largest contributor to the generalization error over the first epochs is $e_3$. This implies that if we do not push the representations to be domain-invariant, there is a hope to get better generalization on unseen domains by focusing on the classifier. Unfortunately, even the basic ERM, without additional loss terms, converges towards training-domain invariance and harms test set separability.

To verify that focusing on the classifier can be beneficial, we take a ResNet-50 pretrained on ImageNet using BYOL algorithm [13] and fix the representations. At this point, the representations of different domains are well distinguishable (similar to the first epochs of a regular training), $e'_1$ for the validation and test domains are 0.123 and 0.062 respectively. This implies that the test domain is more similar to the training domains than the validation domain in BYOL’s space. As we freeze the representations, these numbers serve as lower bounds for the generalization error, and the training can affect only $e_0$ and $e_5$. Experiments show that the ERM baseline with no weight decay (Figs. 3g and 3h), as well as SD algorithm with a large $\beta$, can indeed decrease $e_5$ and approach the lower bound (Tab. 1). GroupDRO and IRM fail to decrease $e_5$, while DANN and ERM+HSIC are not applicable to the setup with fixed representations. This result indicates that avoiding even a little collapse of representations, which in these cases coincides with increased domain invariance, opens new opportunities to decrease the generalization error by targeting just $e_5$. The causal link between poor representations of unseen domains and domain invariance is not clear.

### Model selection
To verify whether model selection using validation domains is suitable, we compute the correlation between accuracy of the models on validation and test domains at each epoch for a given dataset/algorithim pair. On Colored MNIST we get 0.98, 0.8, 0.73 and 0.31 correlations for ERM+HSIC, DeepCORAL, SD and DANN, respectively. On Camelyon17 basically no correlation is observed, which is similar to the findings in [23]. Corresponding plots are shown in Appendix H.

### Model complexity
To test the dependence of generalization failures on model complexity, we followed [26] to train several ResNet 18K models ($k = 2, 4, 8, 16, 32, 64$) with ERM+HSIC objective on Colored MNIST ($\beta = 1$). As seen on Fig. 6d, larger models better fit the training set, but get worse training-test domain invariance ($d'_1$).

## 7. Conclusion
In this paper we have developed tools to examine failure modes of domain generalization algorithms. We have iden-
We have identified two distinct patterns for failures. In one case, most algorithms achieved domain invariance on training domains but not on unseen domains which resulted in large generalization errors. In the other case, we showed that increased domain invariance coincides with degradation of learned representations on unseen domains. We showed evidence that without achieving domain invariance it is possible to target one of the failure modes and improve generalization even with the basic ERM algorithm. We hope these findings will catalyze future work on the development of algorithms with better domain generalization capabilities.

References

[1] Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, and Mario Marchand. Domain-adversarial neural networks. *stat.*, 1050:15, 2014. 6
[2] Ehab A AlBadawy, Ashiribani Saha, and Maciej A Mazurowski. Deep learning for segmentation of brain tumors: Impact of cross-institutional training and testing. *Medical physics*, 45(3):1150–1158, 2018. 2
[3] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019. 1, 2, 6
[4] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 456–473, 2018. 1, 2
[5] P. Bándi, O. Geessink, Q. Manson, M. Van Dijk, M. Balkenhol, M. Hermens, B. Ehteshami Bejnordi, B. Lee, K. Paeng, A. Zhong, Q. Li, F. G. Zanjani, S. Zinger, K. Fukuta, D. Komura, V. Ovtcharov, S. Cheng, S. Zeng, J. Thagaard, A. B. Dahl, H. Lin, H. Chen, L. Jacobsson, M. Hedlund, M. Çetin, E. Halıcı, H. Jackson, R. Chen, F. Both, J. Franke, H. Küsters-Vandevelde, W. Vreuls, P. Bult, B. van Ginneken, J. van der Laak, and G. Litjens. From detection of individual metastases to classification of lymph node status at the patient level: The camelyon17 challenge. *IEEE Transactions on Medical Imaging*, 38(2):550–560, 2019. 6
[6] Dengxin Dai and Luc Van Gool. Dark model adaptation: Semantic image segmentation from daytime to nighttime. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 3819–3824. IEEE, 2018. 2
[7] Samuel Dodge and Lina Karam. A study and comparison of human and deep learning recognition performance under visual distortions. In *2017 26th international conference on computer communication and networks (ICCCN)*, pages 1–7. IEEE, 2017. 2
[8] Tigran Galstyan, Hrant Khachatrian, Greg Ver Steeg, and Aram Galstyan. Robust classification under class-dependent domain shift. *arXiv preprint arXiv:2007.05335*, 2020. 1, 2, 6
[9] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016. 1, 2
[10] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020. 1, 2
[11] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *International conference on algorithmic learning theory*, pages 63–77. Springer, 2005. 2
[12] A. Gretton, A.J. Smola, J. Huang, Marcel Schmittfull, K.M. Borgwardt, Bernhard Schölkopf, J. Candela, M. Sugiyama, A. Schwaighofer, and N. Lawrence. Covariate shift by kernel mean matching. *Dataset Shift in Machine Learning, 131-160 (2009)*, 01 2009. 2, 6
[13] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Ghehselaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent - a new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc., 2020. 8
[14] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *International Conference on Learning Representations*, 2021. 1, 2, 6
[15] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadalvath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv preprint arXiv:2006.16241*, 2020. 2
[16] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019. 2
[17] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 6
[18] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Sara Beery, et al. Wilds: A benchmark of in-the-wild distribution shifts. *arXiv preprint arXiv:2012.07421*, 2020. 2, 6
[19] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. 6
[20] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 1
[21] Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 624–639, 2018. 1, 2
[22] Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. Detecting and correcting for label shift with black box predictors.
In *International conference on machine learning*, pages 3122–3130. PMLR, 2018.

[23] John P Miller, Rohan Taori, Aditi Raghunathan, Shiori Sagawa, Pang Wei Koh, Vaishaal Shankar, Percy Liang, Yair Carmon, and Ludwig Schmidt. Accuracy on the line: On the strong correlation between out-of-distribution and in-distribution generalization. In *International Conference on Machine Learning*, pages 7721–7735. PMLR, 2021.

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

[25] Vaishnavh Nagarajan, Anders Andressen, and Behnam Neyshabur. Understanding the failure modes of out-of-distribution generalization. In *International Conference on Learning Representations*, 2021.

[26] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. In *International Conference on Learning Representations*, 2020.

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

[28] Kuan-Chuan Peng, Ziyan Wu, and Jan Ernst. Zero-shot deep domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 764–781, 2018.

[29] Mohammad Pezeshki, Sékou-Oumar Kaba, Yoshua Bengio, Aaron Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. *arXiv preprint arXiv:2011.09468*, 2020.

[30] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. The MIT Press, 2009.

[31] Alexandre Rame, Corentin Dancette, and Matthieu Cord. Fishr: Invariant gradient variances for out-of-distribution generalization. *arXiv preprint arXiv:2109.02934*, 2021.

[32] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5389–5400. PMLR, 09–15 Jun 2019.

[33] Elan Rosenfeld, Pradeep Kumar Ravikumar, and Andrej Risteski. The risks of invariant risk minimization. In *International Conference on Learning Representations*, 2021.

[34] Marco Saerens, Patrice Latinne, and Christine Decaestecker. Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. *Neural computation*, 14(1):21–41, 2002.

[35] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2019.

[36] Shiori Sagawa*, Pang Wei Koh*, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2020.

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

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

[39] Amos Storkey. When training and test sets are different: characterizing learning transfer. *Dataset shift in machine learning*, 30:3–28, 2009.

[40] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pages 443–450. Springer, 2016.

[41] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE, 2011.

[42] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[43] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018.

[44] Kun Zhang, Bernhard Schölkopf, Krikamol Muandet, and Zhikun Wang. Domain adaptation under target and conditional shift. In *International Conference on Machine Learning*, pages 819–827. PMLR, 2013.

[45] Shanshan Zhao, Mingming Gong, Tongliang Liu, Huan Fu, and Dacheng Tao. Domain generalization via entropy regularization. *Advances in Neural Information Processing Systems*, 33, 2020.
A. Proofs

This appendix includes the proofs of Propositions 1 and 2.

**Proof of Proposition 1.** Let us fix a label value \( y \in \mathcal{Y} \) and two distinct two distinct domains \( p_1(x, y) \) and \( p_2(x, y) \) from the union of test and training domains. Let \( X_1 \sim p_1(x, y), X_2 \sim p_2(x, y), Z_1 = h(X_1), \) and \( Z_2 = h(X_2) \). Let \( w \) be the parameters of the common classifier for all training and test domains (i.e., the classifier that makes \( e_2 = 0 \)). Then for all \( z \),

\[
p_1(z|y) = \frac{p_1(z)p_1(y|z)}{\int p_1(z)p_1(y|z')dz'} = \frac{p_2(z)p_1(y|z)}{\int p_2(z)p_1(y|z')dz'} = \frac{p_2(z)\delta_y(\arg\max_k f_w(z)_k)}{\int p_2(z')\delta_y(\arg\max_k f_w(z')_k)dz'} = \frac{p_2(z)p_2(y|z)}{\int p_2(z)p_2(y|z')dz'} = p_2(z|y).
\]

Above \( \delta_y(y') \) is the Kroeker’s delta function and the last two transitions use the fact the classifier with parameters \( w \) correctly classifies representations of both domains.

**Proof of Proposition 2.** Let \( X_1 \sim p_1^1(x, y), Z_1 = h(X_1) \) be a random sample from the first training domain and \( X_1 \sim p_1^3(x, y), Z_i = h(X_i) \) be a random sample from the \( i \)-th test domain. Let \( w \) be the parameters of the classifier found by the training algorithm that classifies correctly representations of all training domains. If we fix a label \( y \in \mathcal{Y} \), then

\[
\mathbb{P}(\arg\max_k f_w(Z_i)_k = y \mid Y_i = y) = \mathbb{P}(\arg\max_k f_w(Z_1)_k = y \mid Y_1 = y),
\]

as \( p_1^1(z|y) = p_1^3(z|y) \) by the assumptions. As \( e_0 = 0 \) implies that the classifier \( f_w \) perfectly classifies representations from the first training domain, the right-hand-side of (7) will be equal to 1. Therefore, the \( f_w \) will correctly classify also the representations of test domain \( p_1^3(x, y) \). As \( i \) was arbitrary, this concludes the proof.

B. Decompositions

This appendix describes additional properties of the decompositions presented in the main text (see Sec. 4.4).

Let \( \mathcal{D} = \{p_i(x, y)\}_{i=1}^{n_1+n_3} \) be a family of \( n_1 + n_3 \) domains. Let \( S \) be a subset of \( \{1, 2, \ldots, n_1 + n_3\} \) of size \( n_1 \), chosen uniformly at random, and let \( \hat{S} \triangleq \{(1, 2, \ldots, n_1 + n_3) \setminus S\} \) be the complement of \( S \). This subset \( S \) defines a partition of \( \mathcal{D} \) into training and test domains. Let \( w(S) \) and \( \theta(S) \) denote the parameters of the classification head and the feature extractor after training on domains specified by \( S \). Let \( (X_1, Y_1), \ldots, (X_{n_1+n_3}, Y_{n_1+n_3}) \) be random variables drawn from distributions \( p_1(x, y), \ldots, p_{n_1+n_3}(x, y) \), respectively. To simplify the derivations below, we define

\[
F(A; w, \theta) = \frac{1}{|A|} \sum_{i \in A} \mathbb{E}_{X_i, Y_i} \left[ \ell(f_w(h_\theta(X_i)), Y_i) \right],
\]

for any subset \( A \) of \( \{1, 2, \ldots, n_1 + n_3\} \). To avoid unnecessary technical complications in statements and proofs below, we assume that the feature extractor parameters \( \theta \), label classifier parameters \( w \), and domain classifier parameters \( \theta \) belong to compact domains. This will allow us to replace infimum operators in the definitions of generalization and invariance metrics by minimum operators.

With these conventions, the generalization metrics can be written the following way

\[
e_0'(S) = F(S; w(S)), \quad e_1'(S) = \min_{w' \in \mathcal{W}} F(S; w', \theta(S))
\]

\[
e_2'(S) = F(\hat{S}; \hat{w}(S)), \quad \text{where } \hat{w}(S) \in \arg \min_{w' \in \mathcal{W}} F(S \cup \hat{S}; w', \theta(S)),
\]

\[
e_3'(S) = F(\hat{S}; w(S)),
\]

Next, we prove several statements to establish the relationship between the metrics.
Proposition 3. Assume that \( n_1 = n_3 \) and \((w(S), \theta(S)) \in \arg \min_{w', \theta'} F(w', S, \theta')\). Then \( \mathbb{E}_S [e_0'(S)] \leq \mathbb{E}_S [e_1'(S)] \).

Proof. Let the generalization metrics \( e_1'(S) = \min_{w \in \mathcal{W}} F(\bar{S}; w', \theta(S)) \) be defined as above. Then, \( (i) e_1'(S) \leq e_2'(S) \). Furthermore, if \( w(S) \in \arg \min_{w'} F(S; w', \theta(S)) \) then \( (ii) e_2'(S) \leq e_3'(S) \).

Proof. \( (i) \) We have that

\[
e_1'(S) = \min_{w \in \mathcal{W}} F(\bar{S}; w', \theta(S)) \leq F(\bar{S}; \hat{w}(S), \theta(S)) = e_2'(S).
\]

\( (ii) \) Let \( \alpha \overset{\Delta}{=} n_1/(n_1 + n_3) \). As \( \hat{w}(S) \) is a global minimum of \( F(S \cup \bar{S}; w', \theta(S)) \), we have that

\[
F(S \cup \bar{S}; \hat{w}(S), \theta(S)) \leq F(S \cup \bar{S}; \hat{w}(S), \theta(S)),
\]

which is equivalent to

\[
\alpha F(S; \hat{w}(S), \theta(S)) + (1 - \alpha) F(\bar{S}; \hat{w}(S), \theta(S)) \leq \alpha F(S; w(S), \theta(S)) + (1 - \alpha) F(\bar{S}; \hat{w}(S), \theta(S)).
\]

This simplifies to

\[
F(\bar{S}; \hat{w}(S), \theta(S)) \leq F(\bar{S}; w(S), \theta(S)) + \frac{\alpha}{1 - \alpha} \left( F(S; w(S), \theta(S)) - F(S; \hat{w}(S), \theta(S)) \right).
\]

The additional assumption that \( w(S) \in \arg \min_{w'} F(S; w', \theta(S)) \) implies that \( F(S; w(S), \theta(S)) \leq F(S; \hat{w}(S), \theta(S)) \). Connecting this with \( (16) \) we get

\[
F(\bar{S}; \hat{w}(S), \theta(S)) \leq F(\bar{S}; w(S), \theta(S)),
\]

which is the same as \( e_2'(S) \leq e_3'(S) \).

Next we investigate the relationship between invariance metrics. Similar to the function \( F(A; w, \theta) \) defined above, we define

\[
G(A; \omega, \theta) = \frac{1}{|A|} \sum_{i \in A} \mathbb{E}_{X_i} [\ell (g_{\omega}(h_{\theta}(X_i)), i)],
\]

\( 12 \)
for any subset $A$ of $\{1, 2, \ldots, n_1 + n_3\}$. With these conventions, the invariance metrics can be written as follows:

$$d'_2(S) = 1 - \frac{1}{C} \sum_{y=1}^{C} \min_{\omega \in \Omega} E \left[ \frac{1}{n_1 + n_3} \sum_{i \in S \cup \bar{S}} \ell(g_{\omega}(h_{\theta}(X_i)), i) \right] E_y \frac{1}{n_1 + n_3},$$

$$d'_1(S) = 1 - \frac{1}{C} \sum_{y=1}^{C} \min_{\omega \in \Omega} E \left[ \frac{1}{n_1 + n_3} \sum_{i \in S \cup \bar{S}} \ell(g_{\omega}(h_{\theta}(X_i)), i) \right] E_y \frac{1}{n_1 + n_3},$$

where $E_y$ denotes the event $(Y_i = y \land \cdots \land Y_{n_1 + n_3} = y)$.

The relationship between $d'_0(S)$ and $d'_1(S)$ is significantly more complicated as the sets of domains are different, and the accuracy scores are not directly comparable. Furthermore, the training algorithm can be “adversarial” in the sense that its produced representations of training domains are more distinguishable compared to that of testing domains. The relationship between $d'_1(S)$ and $d'_2(S)$ is simpler, and the desired inequality can be proved with mild assumptions.

**Proposition 5.** Let $d'_1(S)$ and $d'_2(S)$ be defined as above. Assuming that $P(Y_i = y) = 1/C$ for all $i \in \{1, 2, \ldots, n_1 + n_3\}$ and $y \in \{1, 2, \ldots, C\}$, we have that $d'_1(S) \leq d'_2(S)$.

**Proof.**

$$d'_2(S) = 1 - \frac{1}{C} \sum_{y=1}^{C} \min_{\omega \in \Omega} E \left[ \frac{1}{n_1 + n_3} \sum_{i \in S \cup \bar{S}} \ell(g_{\omega}(h_{\theta}(X_i)), i) \right] E_y \frac{1}{n_1 + n_3},$$

$$= 1 - \frac{1}{C} \sum_{y=1}^{C} \min_{\omega \in \Omega} \left( \frac{1}{n_1 + n_3} \sum_{i \in S \cup \bar{S}} E_{X_i} \left[ \ell(g_{\omega}(h_{\theta}(X_i)), i) \right] E_y \frac{1}{n_1 + n_3} \right),$$

$$\geq 1 - \min_{\omega \in \Omega} \left( \frac{1}{C} \sum_{y=1}^{C} \frac{1}{n_1 + n_3} \sum_{i \in S \cup \bar{S}} E_{X_i} \left[ \ell(g_{\omega}(h_{\theta}(X_i)), i) \right] E_y \frac{1}{n_1 + n_3} \right),$$

$$= 1 - \min_{\omega \in \Omega} \left( \frac{1}{n_1 + n_3} \sum_{i \in S \cup \bar{S}} E_{X_i} \left[ \ell(g_{\omega}(h_{\theta}(X_i)), i) \right] \right),$$

$$= d'_1(S).$$

\[ \square \]

**C. Dataset Samples**

Fig. 4 shows one sample per class per domain for Colored MNIST and Camelyon17 datasets used in our experiments. Note that Camelyon17 images are originally four-channel RGBA images, while Colored MNIST images have 50 “channels”, the first three of which are interpreted as RGB in the figure.

**D. Hyperparameters of the algorithms**

For all algorithms we fixed the regularization strength hyperparameter space to $\{0.1, 0.5, 1, 2, 5, 10, 15\}$. We used SGD optimizer in all cases. For Colored MNIST dataset we used learning rate of 0.01 and for Camelyon17 we used 0.001. All methods used 0.01 weight decay unless explicitly noted (e.g. in Tab. 1). We trained all our algorithms for fixed 10 epochs. One epoch of training on Camelyon17 took about 30 minutes (some algorithms, e.g. DANN, IRM, take a bit longer because of more complicated computations) on our machine with two NVIDIA Titan V GPUs, while the algorithms on Colored MNIST took only several seconds.

To produce the plots used in this paper we extracted learned representations by forwarding the data through the learned network and trained multiple logistic regression functions. The duration of this process strongly depends on the size of the
representations and the number of samples in the dataset. Processing the entire dataset for one model on a single Titan V GPU takes 20 minutes for Camelyon17 and less than a minute for Colored MNIST. To produce a single plot we process 10 models (after each epoch) or 8 models (for each value of the hyperparameter \( \beta \)). For some plots, we processed 10 models per epoch to see the behavior in more details in the first three epochs (e.g., Fig. 3f).

### E. More Results

In this section we provide more details about the trained models. We had trouble training IRM on both datasets. For many combinations of hyperparameters the loss was getting NaN at early stages of the training. One of the successful attempts was on Colored MNIST with \( \beta = 10 \). The biggest contributor to the error was changing during the training, as shown on Fig. 6a.

Fig. 6 shows generalization errors and domain-distinguishability on Colored MNIST for SD and GroupDRO algorithms. Still, \( e_2 \) and \( e_3 \) are the main contributors to the generalization error. Fig. 6d shows that more complex networks (i.e., ResNets with more parameters) have smaller errors on the training set and less invariance across training and test domains.

Fig. 7 shows the performance of three more algorithms on Camelyon17. With strong enough regularization, the models underfit the training set. The biggest contributor to the generalization error for other models is the test set inseparability. Fig. 8 shows the impact of the random seed when DeepCORAL is trained on Camelyon17. Some variance is visible in both generalization error and domain distinguishability. In all three cases the algorithm increases domain invariance by the end of the training, but it does not translate into better generalization.

### F. Notes on Label Shift

Although label shift is common in real-world applications, it makes error analysis extremely complicated. First, when the number of domains is large, characterizing types of label shift is a challenge on its own. Label shift can appear between training domains, between individual training and test domains, and between the union of training domains and the union of test domains. iWildCam dataset from WILDS benchmark has all of these shifts. Second, the accuracy of the model can change in both directions on the test domains under label shift. In case when the test set contains more examples from “easier” classes, the error on the test set might be low, and it can hide the performance drop due to the domain shift. We also observe that most of the current domain generalization algorithms simply ignore the label shift issue. In fact, the proof of Proposition 1 shows that if \( e_2' = 0 \) and the representations are domain invariant with respect to the union of training and test domains, label distribution \( p(y) \) is also invariant. It implies that enforcing invariance of representations \( p(z) \) when \( p(y) \) is not invariant is not desirable, as a common classifier will not exist. Finally, to the best of our knowledge, robustness of algorithms with respect to unknown label shifts is not explored even if \( p(x|y) \) is constant across domains. Current label shift literature (e.g., [22]) assumes access to the test domain and discusses adaptation strategies.

### G. Visualization of Representation Spaces

To visualize the learned representations, we forward pass the datasets (all domains) through the networks, take the representations \( z = h_\theta(x) \), perform a single two-dimensional Principal Component Analysis for each model (on the combination of all domains), and plot the results. Colors encode the labels, while the marker type encodes the domain. Fig. 9a shows a typical failure on Colored MNIST dataset. For example, the representations of digits 1 are grouped in multiple clusters. Moreover, the representations of the samples from the validation domain \( V_2^1 \) (indicated by star markers) are quite far from the other clusters. This shows that a linear classifier that successfully works on the training domains might not be able to classify the digits in the validation domain. Fig. 9b shows the more successful case obtained with ERM+HSIC algorithm where the representations of digits are grouped regardless of the domains.

Fig. 9c shows the space learned by ERM baseline on Camelyon17 dataset. It demonstrates a case when the dots are clearly separable according to the label (color) for the training domains, but there are a few dots from the validation and test domains (\( V_2^1 \) and \( V_3^1 \)) that are in the neighborhood of the wrong color. On the 2D space it can be seen that the representations of the samples from validation and test domains are not fully separable. Fig. 9d demonstrates the case when the regularization is too strong and the representations of most samples have collapsed near (0, 0), leading to large \( e_0 \).

### H. Correlation Plots

The first row of Fig. 5 shows correlation plots between generalization error \( e_3' \) and training-validation domain distinguishability \( d_1' \) for various algorithms. Every dot corresponds to a single model saved at each epoch. We expect to see positive correlation between these metrics. For Colored MNIST, DeepCORAL shows 0.52 correlation (Fig. 5a), while ERM+HSIC
has 0.05 correlation (Fig. 5b). The latter is explained by large $c_0$ when regularization is too strong. We do not see any correlation between $e'_3$ and $d'_1$ on Camelyon 17. The second row shows the relationship between the errors of many models on validation and test sets. High correlation implies that the accuracy on the validation set can serve as a good metric for model selection. ERM+HSIC, the only algorithm that performed well on Colored MNIST, demonstrates high correlation (Fig. 5e. The correlation for other algorithms is lower. Fig. 5h shows no correlation for DeepCORAL trained on Camelyon17 with various seeds.

Figure 4. Samples of Colored MNIST and Camelyon17 datasets. Both datasets have three training domains $D_1^1, D_1^2, D_1^3$, one validation domain $D_2^1$ and one test domain $D_3^1$. The first row of (b) shows normal tissue samples, while the samples of the second row contain tumor.

Figure 5. First row: relationship between generalization error $e'_3$ and training-validation domain distinguishability $d'_1$ for various algorithms on (a-b) Colored MNIST and (c-d) Camelyon 17 datasets. Colors of the dots indicate the strength of regularization. Second row: correlation plots between validation and test errors. Colors of the dots indicate the strength of regularization for the models trained on Colored MNIST (e-g) or random seed for the models trained on Camelyon17 (h).
Figure 6. Decomposition of generalization errors and domain-distinguishability of three more algorithms on Colored MNIST dataset measured on the validation domains. Horizontal axis corresponds to regularization strength of the algorithms.

Figure 7. Decomposition of generalization errors and domain-distinguishability of three more algorithms on Camelyon17 dataset measured on the validation domains. Horizontal axis corresponds to regularization strength of the algorithms.

Figure 8. The decomposition of generalization errors and domain-distinguishability of three DeepCORAL models trained on Camelyon17 ($\beta = 5$) with different random seeds, measured on the validation domain.
Figure 9. 2D PCA transformations of the learned representations of several models (from the last epoch). Colors indicate the labels, while marker shapes indicate the domains.