A. Acknowledgements

Nanyang Ye was supported by National Natural Science Foundation of China under Grant 62106139, in part by National Key R&D Program of China 2018AAA0101200, in part by National Natural Science Foundation of China under Grant (No. 61829201, 61832013, 61960206002, 62061146002, 42050105, 62032020), in part by the Science and Technology Innovation Program of Shanghai (Grant 18XD1401800), and in part by Shanghai Key Laboratory of Scalable Computing and Systems, and in part by BIREN Tech.

B. Proofs

Proposition 1. *For any non-negative probability functions p and q of two environments, diversity shift $D_{\text{div}}(p, q)$ and correlation shift $D_{\text{cor}}(p, q)$ are always bounded between 0 and 1, inclusively.*

Proof. Apparently, $D_{\text{div}}(p, q)$ and $D_{\text{cor}}(p, q)$ are always non-negative, so we are left to prove the upper bound. By the triangle inequality and that every probability function sums up to one over all possible outcomes, we have

$$D_{\text{div}}(p, q) = \frac{1}{2} \int_{\mathcal{S}} |p(\mathbf{z}) - q(\mathbf{z})| \, d\mathbf{z} \leq \frac{1}{2} \int_{\mathcal{S}} \big[p(\mathbf{z}) + q(\mathbf{z})\big] \, d\mathbf{z} \leq 1. \tag{6}$$

Similarly, we also have

$$\begin{aligned} D_{\text{cor}}(p, q) &= \frac{1}{2} \int_{\mathcal{T}} \sqrt{p(\mathbf{z})\, q(\mathbf{z})} \sum_{\mathbf{y} \in \mathcal{Y}} |p(\mathbf{y} \,|\, \mathbf{z}) - q(\mathbf{y} \,|\, \mathbf{z})| \, d\mathbf{z} \\ &\leq \frac{1}{2} \int_{\mathcal{T}} \sqrt{p(\mathbf{z})\, q(\mathbf{z})} \sum_{\mathbf{y} \in \mathcal{Y}} \big[p(\mathbf{y} \,|\, \mathbf{z}) + q(\mathbf{y} \,|\, \mathbf{z})\big] \, d\mathbf{z} \qquad (7) \\ &= \frac{1}{2} \int_{\mathcal{T}} 2\sqrt{p(\mathbf{z})\, q(\mathbf{z})} \, d\mathbf{z} \leq \frac{1}{2} \int_{\mathcal{T}} \big[p(\mathbf{z}) + q(\mathbf{z})\big] \, d\mathbf{z} \leq 1. \end{aligned}$$

□

Lemma 1. *Suppose there are equal amount of examples from two environments, then for any example $\mathbf{x} \in \mathcal{X}$ sampled from either of the environments, the probability $\alpha(\mathbf{x})$ of a prediction $t(\mathbf{x}) \in \{0, 1\}$ that predicts the sampling environment of $\mathbf{x}$ being correct is*

$$\frac{(1 - t(\mathbf{x})) \cdot p(\mathbf{x}) + t(\mathbf{x}) \cdot q(\mathbf{x})}{p(\mathbf{x}) + q(\mathbf{x})}.$$

Proof. First of all, we need to define several quantities. Let the probability of an example being sampled from the first environment be $P(E = 0)$ and the probability of an example being sampled from the second environment be $P(E = 1)$. Since there are equal amount of examples from both environment, we have $P(E = 0) = P(E = 1) = \frac{1}{2}$. The probability of an example (from one of the environments) taking on a particular value $\mathbf{x}$ is $P(X = \mathbf{x} \,|\, E = 0)$ and $P(X = \mathbf{x} \,|\, E = 1)$. By definition, $P(X = \mathbf{x} \,|\, E = 0) = p(\mathbf{x})$ and $P(X = \mathbf{x} \,|\, E = 1) = q(\mathbf{x})$. The probability of an example taking on a particular value $\mathbf{x}$ (regardless of the environment) is given by

$$\begin{aligned} P(X = \mathbf{x}) &= P(X = \mathbf{x}, E = 0) + P(X = \mathbf{x}, E = 1) \\ &= P(X = \mathbf{x} \,|\, E = 0) \cdot P(E = 0) + P(X = \mathbf{x} \,|\, E = 1) \cdot P(E = 1) = \frac{p(\mathbf{x}) + q(\mathbf{x})}{2}. \end{aligned} \tag{8}$$

The probability of a given example $\mathbf{x}$ being sampled from the first environment is

$$P(E = 0 \,|\, X = \mathbf{x}) = \frac{P(X = \mathbf{x} \,|\, E = 0) \cdot P(E = 0)}{P(X = \mathbf{x})} = \frac{p(\mathbf{x})}{2P(X = \mathbf{x})} = \frac{p(\mathbf{x})}{p(\mathbf{x}) + q(\mathbf{x})}. \tag{9}$$

Similarly, the probability of $\mathbf{x}$ being sampled from the second environment is

$$P(E = 1 \,|\, X = \mathbf{x}) = \frac{q(\mathbf{x})}{p(\mathbf{x}) + q(\mathbf{x})}. \tag{10}$$

Together, the overall probability of some prediction $t(\mathbf{x})$ being correct is

$$\alpha(\mathbf{x}) = (1 - t(\mathbf{x})) \cdot P(E = 0 \,|\, X = \mathbf{x}) + t(\mathbf{x}) \cdot P(E = 1 \,|\, X = \mathbf{x}) = \frac{(1 - t(\mathbf{x})) \cdot p(\mathbf{x}) + t(\mathbf{x}) \cdot q(\mathbf{x})}{p(\mathbf{x}) + q(\mathbf{x})}. \tag{11}$$

□

Theorem 1. *The classification accuracy of a network trained to discriminate two different environments is bounded above by* $\frac{1}{2}\int_{\mathcal{X}} \max\{p(\mathbf{x}), q(\mathbf{x})\}$*, as the data size tends to infinity. This optimal performance is attained only when the following condition holds: for every* $\mathbf{x} \in \mathcal{X}$ *that is* not *i.i.d. in the two environments,* i.e. $p(\mathbf{x}) \neq q(\mathbf{x})$*, there exists some* $\mathbf{y} \in \mathcal{Y}$ *such that* $\hat{p}(\mathbf{y}, \mathbf{z}) \neq \hat{q}(\mathbf{y}, \mathbf{z})$ *where* $\mathbf{z} = g(\mathbf{x})$.

Proof. To formally state that the network attains optimal performance in classifying the environments, we note that for any example $\mathbf{x} \in \mathcal{X}$, the probability $\alpha(\mathbf{x})$ of a prediction $t(\mathbf{x}) \in \{0, 1\}$ being correct is

$$\frac{(1 - t(\mathbf{x})) \cdot p(\mathbf{x}) + t(\mathbf{x}) \cdot q(\mathbf{x})}{p(\mathbf{x}) + q(\mathbf{x})}. \tag{12}$$

This has been shown by Lemma 1. Hence, the overall classification accuracy is given by

$$\mathbb{E}_X[\alpha(\mathbf{x})] = \mathbb{E}_X\left[\frac{(1 - t(\mathbf{x})) \cdot p(\mathbf{x}) + t(\mathbf{x}) \cdot q(\mathbf{x})}{p(\mathbf{x}) + q(\mathbf{x})}\right] = \frac{1}{2}\int_{\mathcal{X}} (1 - t(\mathbf{x})) \cdot p(\mathbf{x}) + t(\mathbf{x}) \cdot q(\mathbf{x}). \tag{13}$$

The best possible classification accuracy that can be attained by the network is determined by the environments alone, which is

$$\frac{1}{2}\int_{\mathcal{X}} \max_t \{(1 - t(\mathbf{x})) \cdot p(\mathbf{x}) + t(\mathbf{x}) \cdot q(\mathbf{x})\} = \frac{1}{2}\int_{\mathcal{X}} \max\{p(\mathbf{x}), q(\mathbf{x})\}. \tag{14}$$

The above maximum is attained by t^* satisfying the following equations for every $\mathbf{x}$ such that $p(\mathbf{x}) \neq q(\mathbf{x})$:

$$t^*(\mathbf{x}) = \begin{cases} 0 & \text{if } p(\mathbf{x}) > q(\mathbf{x}) \\ 1 & \text{if } p(\mathbf{x}) < q(\mathbf{x}) \end{cases}. \tag{15}$$

In other words, t^* predicts the environment from which an example is more likely sampled. Suppose for some $\mathbf{x}$ such that $p(\mathbf{x}) > q(\mathbf{x})$ we have (i) $g(\mathbf{x}) = \mathbf{z}$ and (ii) $\hat{p}(\mathbf{y}, \mathbf{z}) = \hat{q}(\mathbf{y}, \mathbf{z})$ for every $\mathbf{y} \in \mathcal{Y}$, then there must exist some $\mathbf{x}' \neq \mathbf{x}$ such that $p(\mathbf{x}') < q(\mathbf{x}')$ with $f(\mathbf{x}') = f(\mathbf{x})$ and $g(\mathbf{x}') = g(\mathbf{x})$. It follows that $t(\mathbf{x}) = t(\mathbf{x}')$ because $\mathbf{x}$ and $\mathbf{x}'$ both map to the same $\mathbf{y}$ and $\mathbf{z}$, which makes no difference to h. Finally, it is clear to see that $t \neq t^*$ and therefore the network does not attain the optimal performance. □

C. Practical Estimation of Diversity and Correlation Shift

In this section, we provide complete pseudo codes of our estimation methods for diversity and correlation shift, supported by more theoretical justifications.

C.1. Pseudo codes and supporting theoretical justifications

Algorithm 1 Training procedure of feature extractor and environment classifier

Require: Training environment $\mathcal{E}_{\text{tr}}$ and test environments $\mathcal{E}_{\text{te}}$; mini-batch size N; number of training steps T; loss function ℓ.
Ensure: Feature extractor $g : \mathcal{X} \to \mathcal{F}$; environment classifier $h : \mathcal{F} \times \mathcal{Y} \to [0, 1]$.

1: Initialize network parameters;
2: **for** each training step $t \leftarrow 1, \ldots, T$ **do**
3: sample a mini-batch of training examples $\{(\mathbf{x}_i, \mathbf{y}_i, e_i)\}_{i=1}^N$ from $\mathcal{E}_{\text{tr}}$ (indexed by $e_i = 0$) and $\mathcal{E}_{\text{te}}$ (indexed by $e_i = 1$) while ensuring equal sampling probability for the two environments and for every distinct value $\mathbf{y} \in \mathcal{Y}$ in each environment;
4: **for** each example $(\mathbf{x}_i, \mathbf{y}_i, e_i)$ in the mini-batch **do**
5: $\hat{e}_i \leftarrow h(g(\mathbf{x}_i), \mathbf{y}_i)$;
6: compute loss $\ell(\hat{e}_i, e_i)$ and back-propagate gradients;
7: update network parameters by the accumulated gradients, and then reset the gradients.

The training procedure of our feature extractor g and environment classifier h is described in Algorithm 1. Note that in line 3, we use sample reweighting to ensure the class balance in every environment so that the following assumption holds.

Assumption 1. *For every* $\mathbf{y} \in \mathcal{Y}$*,* $p(\mathbf{y}) = q(\mathbf{y}) > 0$*,* i.e.*, there is no label shift.*

Each environment defines a distribution over $\mathcal{X}$. The two environments share the same labeling rule $f : \mathcal{X} \to \mathcal{Y}$. The feature extractor $g : \mathcal{X} \to \mathcal{F}$ maps every input $\mathbf{x} \in \mathcal{X}$ to an d-dimensional feature vector $\mathbf{z} \in \mathcal{F}$. Put it together, the labeling rule f, the feature extractor g together with the two distributions over $\mathcal{X}$ induces two probability functions $\hat{p}$ and $\hat{q}$ over $\mathcal{X} \times \mathcal{Y} \times \mathcal{F}$.

As mentioned in the paper, for a practical estimation of diversity and correlation shift, we first partition $\mathcal{F}$ into

$$\mathcal{S}' := \{\mathbf{z} \in \mathcal{F} \mid \hat{p}(\mathbf{z}) \cdot \hat{q}(\mathbf{z}) = 0\} \quad \text{and} \quad \mathcal{T}' := \{\mathbf{z} \in \mathcal{F} \mid \hat{p}(\mathbf{z}) \cdot \hat{q}(\mathbf{z}) \neq 0\}, \tag{16}$$

and then estimate the shifts by

$$D'_{\text{div}}(\hat{p}, \hat{q}) := \frac{1}{2} \int_{\mathcal{S}'} |\hat{p}(\mathbf{z}) - \hat{q}(\mathbf{z})| \, d\mathbf{z}, \tag{17}$$

$$D'_{\text{cor}}(\hat{p}, \hat{q}) := \frac{1}{2} \int_{\mathcal{T}'} \sqrt{\hat{p}(\mathbf{z})\, \hat{q}(\mathbf{z})} \sum_{\mathbf{y} \in \mathcal{Y}} |\hat{p}(\mathbf{y} \mid \mathbf{z}) - \hat{q}(\mathbf{y} \mid \mathbf{z})| \, d\mathbf{z}, \tag{18}$$

where we have simply replaced (p, q) with their empirical estimates $(\hat{p}, \hat{q})$ and replaced $(\mathcal{S}, \mathcal{T})$ with $(\mathcal{S}', \mathcal{T}')$ in Definition 1. One might notice a slight difference between the definition of $(\mathcal{S}, \mathcal{T})$ and the definition of $(\mathcal{S}', \mathcal{T}')$, which is that $(\mathcal{S}, \mathcal{T})$ are subsets of $\mathcal{Z}_2$, and therefore only contain non-causal features, whereas $(\mathcal{S}', \mathcal{T}')$ are subsets of $\mathcal{F}$, which could contain representations of both $\mathcal{Z}_1$ and $\mathcal{Z}_2$. Fortunately, this is not an issue for two reasons. First, the features in $\mathcal{S}'$ have no shared support in the two environments, *i.e.* $\hat{p}(\mathbf{z}) \cdot \hat{q}(\mathbf{z}) = 0$. Recall that

$$p(\mathbf{z}) \cdot q(\mathbf{z}) \neq 0 \ \wedge \ \forall \mathbf{y} \in \mathcal{Y} : p(\mathbf{y} \mid \mathbf{z}) = q(\mathbf{y} \mid \mathbf{z}) \tag{19}$$

holds for every $\mathbf{z} \in \mathcal{Z}_1$. This suggests that we would have $\hat{p}(\mathbf{z}) \cdot \hat{q}(\mathbf{z}) \neq 0$ for every representation $\mathbf{z}$ of $\mathcal{Z}_1$, and therefore the integral over $\mathcal{S}'$ would exclude these features. Second, (19) also suggests that the term $|\hat{p}(\mathbf{y} \mid \mathbf{z}) - \hat{q}(\mathbf{y} \mid \mathbf{z})|$ would be relatively small for every $\mathbf{y} \in \mathcal{Y}$ and representation $\mathbf{z}$ of $\mathcal{Z}_1$, so the integral over $\mathcal{T}'$ will not be affected by representations of $\mathcal{Z}_1$.

Once g is trained properly, inputs from training and test environments are all processed by g. Then the output features $\mathcal{F}$ are gathered into $\mathcal{F}_{\text{tr}}$ and $\mathcal{F}_{\text{te}}$, which are used for estimating the shifts as in Algorithm 2 below.

Algorithm 2 Estimation of diversity and correlation shift

Require: Features $\mathcal{F}_{\text{tr}}$ and $\mathcal{F}_{\text{te}}$ from training and test environments; importance sampling size M; thresholds ϵ_{div} and ϵ_{cor}.
Ensure: Estimated diversity shift D'_{div}; estimated correlation shift D'_{cor}.

```
# Prepare for the estimation
F ← F_tr ∪ F_te;
scale F to zero mean and unit variance;
ŵ ← fit by KDE the distribution of F;
F'_tr, F'_te ← split F to recover the original partition;
p̂, q̂ ← fit by KDE the distributions of F'_tr and F'_te;

# Estimate diversity shift
D'_div ← 0;
for t ← 1, ..., M do
    z ← sample from ŵ;
    if p̂(z) < ε_div or q̂(z) < ε_div then
        D'_div ← D'_div + |p̂(z) − q̂(z)| / ŵ(z);
D'_div ← D'_div / 2M;

# Estimate correlation shift
D'_cor ← 0;
for each y ∈ Y do
    p̂_y, q̂_y ← fit by KDE the distributions of the subsets of F'_tr and F'_tr that correspond to the inputs with label y;
    for t ← 1, ..., M do
        z ← sample from ŵ;
        if p̂(z) > ε_cor and q̂(z) > ε_cor then
            D'_cor ← D'_cor + |p̂_y(z)√(q̂(z)/p̂(z)) − q̂_y(z)√(p̂(z)/q̂(z))| / ŵ(z);
D'_cor ← D'_cor / 2M|Y|.
```

C.2. Implementation details

Same as the networks on which the algorithms in Section 3.1 are trained, we use MLP for Colored MNIST and ResNet-18 for other datasets as the feature extractors. All ResNet-18 models are pretrained on ImageNet. The feature extractors are optimized by Adam with a fixed learning rate 0.0003 for $T = 2000$ iterations. The batch size N we used is 32 for each environment, and we set the feature dimension $m = 8$. For every random data split, we keep 90% data for training and use the rest 10% data for validation. We choose the models maximizing the accuracy (in predicting the environments) on validation sets. For datasets with multiple training environments and test environments, the network is trained to discriminate all the environments. The loss function ℓ is the cross-entropy loss in our experiments. As for Algorithm 2, the importance sampling size $M = 10000$, and we empirically set the thresholds $\epsilon_{\text{div}} = 1 \times 10^{-12}$ and $\epsilon_{\text{cor}} = 5 \times 10^{-4}$. We use Gaussian kernels for all the KDEs.

D. Discussion on the Convergence of the Hidden Feature

In this section, we investigate the convergence of the features extracted by the neural network. We base our analysis on the neural tangent kernel (NTK) [39, 102]. We focus on the dynamic of the output of the feature extractor instead of the output of the entire neural network. For simplicity, consider a fully-connected neural network with layers numbered from 0 (input) to L (output), each containing $n_0, \cdots, n_{L-1}$, and $n_L = 1$ neurons. The network uses a Lipschitz, twice-differentiable nonlinearity function $\sigma : \mathbb{R} \to \mathbb{R}$ with bounded second derivative. We define the network function by $f(x;\theta) := h^{(L)}(x;\theta)$, where the function $h^{(\ell)} : \mathbb{R}^{n_0} \to \mathbb{R}^{n_\ell}$ and function $g^{(\ell)} : \mathbb{R}^{n_0} \to \mathbb{R}^{n_\ell}$ are defined from the 0-th layer to the L-th layer recursively by

$$\begin{aligned} g^{(0)}(x;\theta) &:= x, \\ h^{(\ell+1)}(x;\theta) &:= \frac{1}{\sqrt{n_\ell}} W^{(\ell)} g^{(\ell)}(x;\theta), \\ g^{(\ell)}(x;\theta) &:= \sigma(h^{(\ell)}(x;\theta)). \end{aligned}$$

We refer to the output of the $(L-1)$-th layer as the extracted feature, *i.e.* $h^{(L-1)}(x;\theta)$.[3] Given a training dataset $\{(x_i, y_i)\}_{i=1}^n \subset \mathbb{R}^{n_0} \times \mathbb{R}$, consider training the neural network by minimizing the loss function over training data by gradient descent: $\sum_{i=1}^n loss(f(x_i;\theta), y_i)$.

Theorem 2. *Assume that the non-linear activation σ is Lipschitz continuous, twice differentiable with bounded second order derivative. As the width of the hidden layers increase to infinity, sequentially, the hidden layer output $h^{(L-1)}(x;\theta)$ converges to the solution of the differential equation*

$$\begin{aligned} \frac{du^{(L-1,1)}(t)}{dt} &= -H^{(L-1,1)} G^{(L-1,1)}(t), \\ &\vdots \\ \frac{du^{(L-1,n_{L-1})}(t)}{dt} &= -H^{(L-1,n_{L-1})} G^{(L-1,n_{L-1})}(t). \end{aligned}$$

where $u^{(\cdot,\cdot)}(t)$ is the vectorized form of $h^{(L-1)}(x;\theta)$, $H^{(\cdot,\cdot)}$ is the neural tangent kernel corresponding to hidden layer output, and $G^{(\cdot,\cdot)}(t)$ is the vectorized form of the derivative of the loss function corresponding to the hidden later output.

Proof. Let $\theta^{(\ell)}$ denote the parameters in the first ℓ layers. Then the parameters $\theta^{(\ell)}$ evolve according to the differential equation

$$\begin{aligned} \frac{d\theta^{(\ell)}(t)}{dt} &= -\nabla_{\theta^{(\ell)}(t)} loss(f(x_i;\theta(t)), y_i) \\ &= -\sum_{i=1}^n \left[\frac{\partial h^{(\ell)}(x_i;\theta(t))}{\theta^{(\ell)}(t)}\right]^T \nabla_{h^{(\ell)}(x_i;\theta(t))} loss(f(x_i;\theta(t)), y_i), \end{aligned}$$

[3] The output of any of the intermediate layer can be regarded as the extracted feature. Our analysis can be easily extend to other scenarios.

where $t \geq 0$ is the continuous time index, which is commonly used in the analysis of the gradient descent with infinitesimal learning rate. The evolution of the feature $h^{(L-1)}(x_j;\theta)$ of the input x_j, $j \in [n]$ can be written as

$$\frac{dh^{(L-1)}(x_j;\theta(t))}{dt} = -\sum_{i=1}^{n} \frac{\partial h^{(L-1)}(x_j;\theta(t))}{\theta^{(L-1)}(t)} \left[\frac{\partial h^{(L-1)}(x_i;\theta(t))}{\theta^{(L-1)}(t)}\right]^T \nabla_{h^{(L-1)}(x_i;\theta(t))} loss(f(x_i;\theta(t)), y_i).$$

Let $G^{(L-1,k)}(t) = (\nabla_{h^{(L-1,k)}(x_j;\theta(t))} loss(f(x_j;\theta(t)), y_j))_{j\in[n]}$ denote the gradient of the loss function corresponding to the k-th output of the $(L-1)$-th intermediate layer at time t, and $u^{(L-1,k)}(t) = (h^{(L-1,k)}(x_j;\theta(t)))_{j\in[n]}$ denote the k-th output of the ℓ-th intermediate layer at time t, respectively. The evolution of the feature can be written more compactly as

$$\frac{du^{(L-1,k)}(t)}{dt} = -H^{(L-1,k)}(t)G^{(L-1,k)}(t),$$

where $H^{(L-1,k)}(t)$ is the matrix defined as

$$[H^{(L-1,k)}(t)]_{a,b} = \left\langle \frac{\partial h^{(L-1,k)}(x_a;\theta(t))}{\partial \theta^{(L-1)}(t)}, \frac{\partial h^{(L-1,k)}(x_b;\theta(t))}{\partial \theta^{(L-1)}(t)} \right\rangle.$$

Applying Theorem 1 and Theorem 2 in [39], as the width of the hidden layers $n_1, \cdots, n_{L-1} \to \infty$, sequentially, we have $H^{(L-1,k)}(t)$ converges to a fixed kernel $H^{(L-1,k)}$. Thus, the extracted feature (*i.e.* the output of the $(L-1)$-th layer) converges to the solution of the system of the differential equations below

$$\begin{aligned}
\frac{du^{(L-1,1)}(t)}{dt} &= -H^{(L-1,1)}G^{(L-1,1)}(t), \\
&\vdots \\
\frac{du^{(L-1,n_{L-1})}(t)}{dt} &= -H^{(L-1,n_{L-1})}G^{(L-1,n_{L-1})}(t).
\end{aligned}$$

□

E. Effects of the Neural Network Architecture

We have tested neural network architecture (MLP) with different number of parameters on Colored MNIST, as shown in Tab. 4 (estimated values), Tab. 5 (t-test) and Tab. 6 (different number of training epochs). The results demonstrate no significant difference in the effect of various neural network architectures on the estimation of diversity and correlation shift. The results tend to converge as model capacity increases. We also compared with other types of architectures (*e.g.*, EfficientNet) on more complicated datasets, PACS, in Tab. 7, where we have observed similar trends. The network architecture can have effects on the estimation, but we expect the architecture to have enough expressive power.

Network	# Params	Type	Split 0	Split 1	Split 2	Split 3	Split 4
MLP (dim=16)	0.0067M	D_{div}	0.0001	0.0001	0.0002	0.0001	0.0001
		D_{cor}	0.8280	0.7838	0.7752	0.8004	0.7727
MLP (dim=32)	0.0140M	D_{div}	0.0001	0.0002	0.0002	0.0002	0.0000
		D_{cor}	0.7067	0.7055	0.7011	0.4299	0.6807
MLP (dim=64)	0.0299M	D_{div}	0.0000	0.0000	0.0001	0.0000	0.0000
		D_{cor}	0.6901	0.7279	0.7096	0.7366	0.7195
MLP (dim=200)	0.1205M	D_{div}	0.0000	0.0003	0.0000	0.0000	0.0000
		D_{cor}	0.6780	0.5684	0.6628	0.6679	0.6305
MLP (dim=390)	0.3089M	D_{div}	0.0000	0.0000	0.0000	0.0000	0.0000
		D_{cor}	0.6971	0.6788	0.6680	0.7343	0.6846
MLP (dim=1024)	1.4603M	D_{div}	0.0000	0.0000	0.0000	0.0000	0.0000
		D_{cor}	0.6609	0.6639	0.6701	0.6765	0.6248
ResNet-18	11.1724M	D_{div}	0.0000	0.0000	0.0000	0.0000	0.0001
		D_{cor}	0.6677	0.6833	0.6802	0.5834	0.5809

Table 4. Estimated diversity and correlation shift of Colored MNIST on networks with different capacity.

Network	# Params	MLP (dim=16)	MLP (dim=32)	MLP (dim=64)	MLP (dim=200)	MLP (dim=390)	MLP (dim=1024)	ResNet-18
MLP (dim=16)	0.0067M	1.0000	0.6740	0.8348	0.7189	0.7799	0.7035	0.6581
MLP (dim=32)	0.0140M	0.6740	1.0000	0.8277	0.9480	0.8832	0.9639	0.9852
MLP (dim=64)	0.0299M	0.8348	0.8277	1.0000	0.8777	0.9428	0.8614	0.8116
MLP (dim=200)	0.1205M	0.7189	0.9480	0.8777	1.0000	0.9343	0.9838	0.9325
MLP (dim=390)	0.3089M	0.7799	0.8832	0.9428	0.9343	1.0000	0.9180	0.8672
MLP (dim=1024)	1.4603M	0.7035	0.9639	0.8614	0.9838	0.9180	1.0000	0.9485
ResNet-18	11.1724M	0.6581	0.9852	0.8116	0.9325	0.8672	0.9485	1.0000

Table 5. T-test on the estimated values between different architectures on Colored MNIST.

Network	# Epochs	Type	Split 0	Split 1	Split 2	Split 3	Split 4
EfficientNet-b0 (4.67M)	500	D_{div}	0.0000	0.0000	0.0000	0.0000	0.0000
		D_{cor}	0.7667	0.5177	0.6465	0.1567	0.1690
EfficientNet-b0 (4.67M)	1000	D_{div}	0.0045	0.0038	0.0016	0.0036	0.0047
		D_{cor}	0.9172	0.8682	0.8099	0.9293	0.9158
EfficientNet-b0 (4.67M)	2000	D_{div}	0.0028	0.0023	0.0016	0.0022	0.0020
		D_{cor}	0.8350	0.7874	0.8428	0.8871	0.8819
EfficientNet-b0 (4.67M)	4000	D_{div}	0.0007	0.0010	0.0018	0.0005	0.0020
		D_{cor}	0.9248	0.9204	0.8211	0.8516	0.9763

Table 6. Different number of training epochs on Colored MNIST.

Network	# Params	Type	Split 0				Split 1				Split 2				Split 3				Split 4			
			0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
ResNet-18	11.1724	D_{div}	0.9655	0.7852	0.8916	0.9644	0.8106	0.9188	0.8527	0.7553	0.7339	0.9283	0.8325	0.5616	0.9660	0.8540	0.7791	0.7075	0.8183	0.6170	0.8181	0.8061
		D_{cor}	0.0000	0.0001	0.0026	0.0000	0.0016	0.0011	0.0024	0.0000	0.0015	0.0000	0.0003	0.0000	0.0008	0.0000	0.0013	0.0000	0.0000	0.0000	0.0004	0.0000
EfficientNet-b0	4.6676	D_{div}	0.9169	0.7012	0.6586	0.9446	0.8742	0.9897	0.9708	1.5537	1.0244	0.9112	0.9031	0.5640	0.8347	0.9876	1.0300	0.6176	0.7596	1.1299	0.9945	1.2978
		D_{cor}	0.0000	0.0012	0.0004	0.0000	0.0000	0.0000	0.0000	0.0000	0.0003	0.0001	0.0023	0.0000	0.0001	0.0003	0.0005	0.0000	0.0007	0.0007	0.0003	0.0000
EfficientNet-b3	11.4873	D_{div}	0.7642	0.9262	0.9982	1.1570	0.7940	0.8942	0.9789	0.9755	0.6751	1.0670	1.0302	0.8345	1.0010	0.5790	0.9349	0.6927	0.8667	1.1565	0.8798	2.6791
		D_{cor}	0.0000	0.0000	0.0002	0.0000	0.0001	0.0000	0.0182	0.0000	0.0004	0.0004	0.0000	0.0000	0.0005	0.0000	0.0015	0.0000	0.0000	0.0003	0.0000	0.0000
EfficientNet-b5	29.3940	D_{div}	0.8967	0.8822	0.9572	0.8732	0.8544	0.8792	1.1918	0.4856	0.6623	0.8086	0.7450	1.0365	0.5031	0.7685	1.1752	0.5001	0.8214	0.5249	1.0050	0.6596
		D_{cor}	0.0100	0.0001	0.0000	0.0000	0.0002	0.0015	0.0001	0.0000	0.0013	0.0003	0.0041	0.0000	0.0048	0.0046	0.0047	0.0000	0.0011	0.0011	0.0025	0.0

Table 7. Different architectures on PACS.

F. Estimation Results with Error Bars

The table below lists all the results that have been plotted in Figure 3 with standard error bars. The statistics are averaged over five runs of different weight initializations and training/validation splits.

Dataset	Div. shift	Cor. shift
i.i.d. data	0.00 ± 0.00	0.00 ± 0.00
PACS	0.81 ± 0.05	0.00 ± 0.00
Office-Home	0.17 ± 0.02	0.00 ± 0.00
Terra Incognita	0.92 ± 0.06	0.00 ± 0.00
Camelyon	1.07 ± 0.80	0.00 ± 0.00
DomainNet	0.43 ± 0.03	0.08 ± 0.00
Colored MNIST	0.00 ± 0.00	0.55 ± 0.13
CelebA	0.02 ± 0.02	0.29 ± 0.04
NICO	0.11 ± 0.06	0.24 ± 0.08
ImageNet-A	0.02 ± 0.00	0.06 ± 0.05
ImageNet-R	0.06 ± 0.01	0.21 ± 0.01
ImageNet-V2	0.01 ± 0.01	0.49 ± 0.10

Table 8. Estimation of diversity and correlation shift.

G. Datasets

Our benchmark includes the following datasets dominated by diversity shift:

- **PACS** [46] is a common DG benchmark. The datasets contain images of objects and creatures depicted in different styles, which are grouped into four domains, $\{\text{photos}, \text{art}, \text{cartoons}, \text{sketches}\}$. In total, it consists of $9,991$ examples of dimension $(3, 224, 224)$ and 7 classes.

- **OfficeHome** [91] is another common DG benchmark similar to PACS. It has four domains: $\{\text{art}, \text{clipart}, \text{product}, \text{real}\}$, containing $15,588$ examples of dimension $(3, 224, 224)$ and 65 classes.

- **Terra Incognita** [14] contains photographs of wild animals taken by camera traps at different locations in nature, simulating a real-world scenario for OoD generalization. Following DomainBed [31], our version of this dataset only utilize four of the camera locations, $\{\text{L100}, \text{L38}, \text{L43}, \text{L46}\}$, covering $24,788$ examples of dimension $(3, 224, 224)$ and 10 classes.

- **Camelyon17-WILDS** [42] is a patch-based variant of the Camelyon17 dataset [18] curated by WILDS [42]. The dataset contains histopathological image slides collected and processed by different hospitals. Data variation among these hospitals arises from sources like differences in the patient population or in slide staining and image acquisition. It contains $455,954$ examples of dimension $(3, 224, 224)$ and 2 classes collected and processed by 5 hospitals.

On the other hand, these datasets are dominated by correlation shift:

- **Colored MNIST** [9] is a variant of the MNIST handwritten digit classification dataset [45]. The digits are colored either red or green in a way that each color is strongly correlated with a class of digits. The correlation is different during training and test time, which leads to spurious correlation. Following IRM [9], this dataset contains $60,000$ examples of dimension $(2, 14, 14)$ and 2 classes.

- **NICO** [33] consists of real-world photos of animals and vehicles captured in a wide range of contexts such as "in water", "on snow" and "flying". There are 9 or 10 different contexts for each class of animal and vehicle. Our version of this dataset simulates a scenario where animals and vehicles are spuriously correlated with different contexts. More specifically, we make use of both classes appeared in four overlapped contexts: "on snow", "in forest", "on beach" and "on grass" to construct training and test environments (as in Appendix G) that are similar to the setting of Colored MNIST. In total, our split consists of $4,080$ examples of dimension $(3, 224, 224)$ and 2 classes.

Environment	Class	on snow	in forest	on beach	on grass
Training 1	Animal	10	400	10	400
	Vehicle	400	10	400	10
Training 2	Animal	20	390	20	390
	Vehicle	390	20	390	20
Validation	Animal	50	50	50	50
	Vehicle	50	50	50	50
Test	Animal	90	10	90	10
	Vehicle	10	90	10	90

Table 9. Environment splits of NICO and the number of examples in each group.

- **CelebA** [54] is a large-scale face attributes dataset with more than 200K celebrity images, each with 40 attribute annotations. It has been widely investigated in AI fairness studies [21, 72, 98] as well as OoD generalization research [70, 79]. Similar to the setting proposed by [79], our version treats "hair color" as the classification target and "gender" as the spurious attribute. We consider a subset of 27,040 images divided into three environments, simulating the setting of Colored MNIST (where there is large correlation shift). We make full use of the group (blond-hair males) that has the least number of images. See Tab. 10 for more details regarding the environment splits.

Environment	Class	Male	Female
Training 1	blond	462	11,671
	not blond	11,671	462
Training 2	blond	924	11,209
	not blond	11,209	924
Test	blond	362	362
	not blond	362	362

Table 10. Environment splits of CelebA and the number of examples in each group.

H. Model Selection Methods

Among the three commonly-used model selection methods we used, *training-domain validation* and *test-domain validation* are concisely described in [31] as follows:

- **Training-domain validation.** We split each training domain into training and validation subsets. We train models using the training subsets, and choose the model maximizing the accuracy on the union of validation subsets. This strategy assumes that the training and test examples follow a similar distribution.
- **Test-domain validation.** We choose the model maximizing the accuracy on a validation set that follows the distribution of the test domain. We allow one query (the last checkpoint) per choice of hyperparameters, disallowing early stopping.

We use *OoD validation* in place of *leave-one-domain-out validation* (another method employed by [31]) out of two considerations: (i) the Camelyon17 dataset is an official benchmark listed in WILDS [42], which inherently comes with an OoD validation set; (ii) given k training domains, leave-one-domain-out validation is computationally costly (especially when k is large), increasing the number of experiments by $k-1$ times. Moreover, when k is small (*e.g.* $k=2$ in Colored MNIST), the leave-one-domain-out validation method heavily reduces the number of training examples accessible to the models.

- **OoD validation.** We choose the model maximizing the accuracy on a validation set that follows *neither* the distribution of the training domain or the test domain. This strategy assumes that the models generalizing well on the OoD validation set also generalize well on the test set.

I. ImageNet-V2 Experiment

Here we include the experiment results on ImageNet-V2 as an example for datasets exhibiting real-world distribution shift. Experimental comparisons and discussions on ImageNet and its variants are not included in the main paper along with other datasets for three main reasons: **(i)** ImageNet is seldom considered in DG literature; **(ii)** the ImageNet variants are released as validation sets and the data size is relatively small (e.g., 10 images per class for ImageNet-V2); and **(iii)** most of the OoD generalization algorithms assume multiple training domains, however, there is no standard way to construct a multi-domain ImageNet. Hence, we conduct experiments with the algorithms that do not make the multi-domain assumption on ImageNet (as training domain) and ImageNet-V2 (as test domain). These algorithms are CORAL, SagNet, and RSC. As previously shown in Table 1, they are superior or equivalent to ERM in terms of performance on datasets dominated by diversity shift. In comparison, the dominant shift between ImageNet and ImageNet-V2 is the correlation shift. The experiment result on ImageNet-V2 is shown in the table below. As expected, ERM outperforms the other algorithms. Note that the relative ranking of the algorithms is reversed from that in Table 1.

ERM	CORAL	SagNet	RSC
32.4 ± 0.2	31.9 ± 0.5	31.0 ± 0.4	28.3 ± 1.4

Table 11. Performance of ERM and OoD generalization algorithms on ImageNet-V2.

J. Hyperparameter Search Space

For convenient comparison, we follow the search space proposed in [31] whenever applicable.

Condition	Hyperparameter	Default value	Random distribution
ResNet	learning rate	0.00005	$10^{\text{Uniform}(-5,-3.5)}$
	batch size	32	$2^{\text{Uniform}(3,5.5)}$
	batch size (if CelebA)	48	$2^{\text{Uniform}(4.5,6)}$
	batch size (if ARM)	8	8
	ResNet dropout	0	0
	generator learning rate	0.00005	$10^{\text{Uniform}(-5,-3.5)}$
	discriminator learning rate	0.00005	$10^{\text{Uniform}(-5,-3.5)}$
	weight decay	0	$10^{\text{Uniform}(-6,-2)}$
	generator weight decay	0	$10^{\text{Uniform}(-6,-2)}$
MLP	learning rate	0.001	$10^{\text{Uniform}(-4.5,-3.5)}$
	batch size	64	$2^{\text{Uniform}(3,9)}$
	generator learning rate	0.001	$10^{\text{Uniform}(-4.5,-2.5)}$
	discriminator learning rate	0.001	$10^{\text{Uniform}(-4.5,-2.5)}$
	weight decay	0	0
	generator weight decay	0	0
IRM	lambda	100	$10^{\text{Uniform}(-1,5)}$
	iterations annealing	500	$10^{\text{Uniform}(0,4)}$
	iterations annealing (if CelebA)	500	$10^{\text{Uniform}(0,3.5)}$
VREx	lambda	10	$10^{\text{Uniform}(-1,5)}$
	iterations annealing	500	$10^{\text{Uniform}(0,4)}$
	iterations annealing (if CelebA)	500	$10^{\text{Uniform}(0,3.5)}$
Mixup	alpha	0.2	$10^{\text{Uniform}(0,4)}$
GroupDRO	eta	0.01	$10^{\text{Uniform}(-1,1)}$
MMD	gamma	1	$10^{\text{Uniform}(-1,1)}$
CORAL	gamma	1	$10^{\text{Uniform}(-1,1)}$
MTL	ema	0.99	RandomChoice($[0.5, 0.9, 0.99, 1]$)
DANN	lambda	1.0	$10^{\text{Uniform}(-2,2)}$
	disc weight decay	0	$10^{\text{Uniform}(-6,2)}$
	discriminator steps	1	$2^{\text{Uniform}(0,3)}$
	gradient penalty	0	$10^{\text{Uniform}(-2,1)}$
	Adam β_1	0.5	RandomChoice($[0, 0.5]$)
MLDG	beta	1	$10^{\text{Uniform}(-1,1)}$
RSC	feature drop percentage	1/3	Uniform$(0, 0.5)$
	batch drop percentage	1/3	Uniform$(0, 0.5)$
SagNet	adversary weight	0.1	$10^{\text{Uniform}(-2,1)}$
ANDMask	tau	1	Uniform$(0.5, 1.0)$
IGA	penalty	1,000	$10^{\text{Uniform}(1,5)}$
ERDG	discriminator learning rate	0.00005	$10^{\text{Uniform}(-5,-3.5)}$
	T' learning rate	0.000005	$10^{\text{Uniform}(-6,-4.5)}$
	T learning rate	0.000005	$10^{\text{Uniform}(-6,-4.5)}$
	adversarial loss weight	0.5	$10^{\text{Uniform}(-2,0)}$
	entropy regularization loss weight	0.01	$10^{\text{Uniform}(-4,-1)}$
	cross-entropy loss weight	0.05	$10^{\text{Uniform}(-3,-1)}$

Table 12. Hyperparameters, their default values and distributions for random search.