## A. An implementation of AIL

**Algorithm 1** Adversarial Invariant Learning

Input: $N_{\text{pretrain}}$: number of epochs for VAE pre-training, $N_{\text{epoch}}$: number of epochs for training VAE and classifier, $S$: size of dataset, $T$: number of clusters, $\eta$: learning rate for the classifier, $\eta_{adv}$: adversarial learning rate for the VAE.
Initialization
Pre-train VAE with the ELBO loss in Equation 4 for $N_{\text{pretrain}}$ Epochs
**for** $iter = 0$ to $N_{\text{epoch}} - 1$ **do**
  **for** $i = 1$ to $S$ **do**
    For input data batch $\mathbf{x}_i$, generate the latent code batch $g_i$ with VAE encoder part:

$$g_i = \text{VAE}_{\text{encoder}}(\mathbf{x}_i) \tag{10}$$

    Calculate the probability of $\mathbf{x}_i$ belonging to cluster $t$ using Eq. 5.
    **Minimization step**:
    Update the classifier's parameters with stochastic gradient descent to minimize the loss in Eq. 9:

$$\begin{aligned}\Phi \leftarrow \Phi - \eta\nabla_{\Phi}(&\sum_{t=1}^{T}\alpha_{\text{VAE}}(t|\mathbf{x}_i^t)\cdot\ell(h_{\Phi}(\mathbf{x}_i^t),y_i^t)\\&+\lambda\sum_{t=1}^{T}\|\alpha_{\text{VAE}}(t|\mathbf{x}_i^t)\nabla_{w|w=1}\ell(w\cdot h_{\Phi}(\mathbf{x}_i^t),\mathbf{y}_i^t)\|^2)\end{aligned} \tag{11}$$

    **Maximization step**:
    Update the VAE's parameters with SGD to maximize the invariance loss in Eq. 9:

$$\begin{aligned}\theta_{\text{VAE}} \leftarrow \theta_{\text{VAE}} + \eta_{adv}\nabla_{\theta_{\text{VAE}}}(&\sum_{t=1}^{T}\alpha_{\text{VAE}}(t|\mathbf{x}_i^t)\ell(h_{\Phi}(\mathbf{x}_i^t),y_i^t)\\&+\sum_{t=1}^{T}\|\alpha_{\text{VAE}}(t|\mathbf{x}_i^t)\nabla_{w|w=1}\ell(w\cdot h_{\Phi}(\mathbf{x}_i^t),\mathbf{y}_i^t)\|^2)\end{aligned} \tag{12}$$

  **end for**
**end for**
collect the trained classifier $\Phi$

## B. Details of IRM convergence experiment

We consider the following structural equation model (SEM):

$$x \leftarrow e * \text{Gaussian}(0,1) \tag{13}$$
$$y \leftarrow x + e * \text{Gaussian}(0,1) \tag{14}$$
$$z \leftarrow y \tag{15}$$

in which the task is to regress the target $y$ from observable variables $x$ and $z$, where $e \in [0,1]$ is the environment index. In the common setting for OoD generalization research, the index $e$ must known a prior to achieve moderate success in generalizing to unseen domains. However, that is not realisitc in many scenarios. To make IRM work in this setting, we randomly assign a pseudo environment index for each data generated from the SEM $\hat{e}$. The specification of $\hat{e}$ compared to $e$ can cause serious convergence problems for IRM as shown in the Figure 2.

## C. Proof of theorems

**Theorem 3.** ***Backdoor adjustment.*** *Given the structured causal graph in Figure 2 and neural network parameters* $\boldsymbol{\theta}$, $p(y|do(x);\boldsymbol{\theta}) = \sum_t p(y|x, S = f(x,t);\boldsymbol{\theta})p(t)$.

*Proof.* Our proof largely follows [29]. From [23], there are a few rules for do-calculus in deriving causal inference. Given a causal directed acyclic graph $\mathcal{G}$ with nodes $X$, $Y$, $T$, and $S$ and the lower cases $x$, $y$, $t$, $s$ to be the realizations of the nodes' values. We denote $\mathcal{G}_{\overline{X}}$ for the causal graph where all in arrows to $X$ are deleted, and $\mathcal{G}_{\underline{X}}$ where all out arrows from $X$ are deleted. Then the following rules hold:

1. **Rule 1:** Insertion/deletion of observations: $p(y|do(x), t, s) = p(y|do(x), s)$, if $(Y \perp\!\!\!\perp Z|X, W)_{\mathcal{G}_{\overline{X}}}$.
2. **Rule 2:** Action/observation exchange: $p(y|do(x), do(t), s) = p(y|do(x), t, s)$, if $(Y \perp\!\!\!\perp T|X, S)_{\mathcal{G}_{\overline{X}\underline{T}}}$
3. **Rule 3:** Insertion/deletion of actions: $p(y|do(x), do(t), s) = p(y|do(x), s)$, if $(Y \perp\!\!\!\perp T|X, S)_{\mathcal{G}_{\overline{XT(S)}}}$

Then, based on the structured causal graph defined in Figure 2, we have:

$$p(y|do(\mathbf{x})) = \sum_t p(y|do(\mathbf{x}), t)p(t|do(\mathbf{x})) \tag{16}$$

$$= \sum_c p(y|do(\mathbf{x}), t)p(t) \tag{17}$$

$$= \sum_c p(y|\mathbf{x}, t)p(t) \tag{18}$$

$$= \sum_c p(y|\mathbf{x}, t, S)P(S|\mathbf{x}, t)p(t) \tag{19}$$

$$= \sum_c p(y|\mathbf{x}, t, S = f(\mathbf{x}, t))p(t) \tag{20}$$

$$= \sum_c p(y|\mathbf{x}, S = f(\mathbf{x}, t))p(t) \tag{21}$$

The first equation is because of total probability formula. The second equation is because of Rule 1. The third equation is because of Rule 2. The fourth equation is because of conditional probability formula. The fifth equation is because of $S = f(x,t)$. The final equation is because of Rule 1.

□

**Theorem 4.** ***(Lower bound of likelihood).*** *If Assumption 1 holds and the Rényi divergence between* $p_{train}(t)$ *and* $p(t)$ *satisfies:* $D_\infty\left(p_{train}(t)||p(t)\right) \leq C$, *then:*

$$\log p(y|do(x);\boldsymbol{\theta})] = \log \sum_t p(y|x, S = f(x,t);\boldsymbol{\theta})p(t) \geq \min_{\mathcal{P}} \left( \exp^C \sum_{k=1}^{K} \frac{\sum_t \log p(y|x, S = f(x,t);\boldsymbol{\theta})p^k_{train}(t)}{K \sum_t p^k_{train}(t)} \right) \tag{22}$$

*Proof.* The proof of theorem is given below, from Jetson inequality:

$$\log(\sum_t p(y|x, S = f(x,t);\boldsymbol{\theta})p(t)) \geq \sum_t p(t) \log p(y|x, S = f(x,t);\boldsymbol{\theta}) \tag{23}$$

Then as the Rényi divergence between $p_{train}(t)$ and $p(t)$ satisfies: $D_\infty\left(p_{train}(t)||p(t)\right) \leq C$, we have:

$$D_\infty\left(p_{train}(t)||p(t)\right) = \log\left(\sup_t \left(\frac{p_{train}(t)}{p(t)}\right)\right) \leq C \tag{24}$$

Then, we have $p(t) \geq \frac{p_{train}(t)}{\exp^C}$. We insert this equation into the above inequality and obtain:

$$\log(\sum_t p(y|x, S = f(x,t);\boldsymbol{\theta})p(t)) \geq \sum_t p(t) \log p(y|x, S = f(x,t);\boldsymbol{\theta}) \tag{25}$$

$$\geq \frac{1}{\exp^C} \sum_t p_{train}(t) \log p(y|x, S = f(x,t);\boldsymbol{\theta}) \tag{26}$$

From Assumption 1, we know that

$$p_{train}(t) = \frac{1}{K}\sum_{k} p^k_{train}(t) \tag{27}$$

We denote the series of $p^k_{train}(t)$ as $\mathcal{P}$. Then, we obtain:

$$\log(\sum_{t} p(y|x, S = f(x,t); \boldsymbol{\theta})p(t)) \geq \sum_{t} p(t) \log p(y|x, S = f(x,t); \boldsymbol{\theta}) \tag{28}$$

$$\geq \frac{1}{\exp^C} \sum_{t} p_{train}(t) \log p(y|x, S = f(x,t); \boldsymbol{\theta}) \tag{29}$$

$$= \frac{1}{\exp^C} \sum_{t} \frac{1}{K} \sum_{k} p^k_{train}(t) \log p(y|x, S = f(x,t); \boldsymbol{\theta}) \tag{30}$$

$$\geq \min_{\mathcal{P}} \sum_{t} \sum_{k} \frac{p^k_{train}(t) \log p(y|x, S = f(x,t); \boldsymbol{\theta})}{exp^C K} \tag{31}$$

This result shows that if we maximize $\min_{\mathcal{P}} \sum_t \sum_k \frac{p^k_{train}(t) \log p(y|x,S=f(x,t);\boldsymbol{\theta})}{exp^C K}$, then we guarantee that the LHS has a lower bound. □

## D. Module choice

In this section, we will compare different implementations of AIL modules. We first compare a Gaussian mixture model (GMM) and K-means clustering method for clustering. Next, we compare the two different regularization methods for AIL *i.e.*, invariant risk minimization (IRM) or risk extrapolation (REx). The results are shown in Table 5. From Table 5, we can see that using GMM or REx generally lead to degenerated performances. For the degenerated performance of GMM, the reason might be GMM assumes data are generated from the mixture of Gaussian distributions for clustering while from the visualization in the ablation study section, the distribution's shape is different. For the degenerated performance of REx, we speculate that regularizing the variances of risks on different environments does not guarantee that an invariant predictor can be learned.

Table 5. Ablation study: module choices for AIL.

| Test accuracy under clustering modules | | |
|---|---|---|
| Module | K-means | GMM |
| $p = 0.15$ | 74.3± 0.9% | **77.3 ± 1.9%** |
| $p = 0.2$ | **65.1± 1.9%** | 65.0 ± 1.1% |
| $p = 0.25$ | **66.8 ± 3.3%** | 57.5 ± 3.5% |
| $p = 0.3$ | **65.1± 3.4%** | 59.2 ± 4.8% |

| Test accuracy under regularization methods | | |
|---|---|---|
| Methods | IRM | REx |
| $p = 0.15$ | **74.3± 0.9%** | 72.5 ± 1.3% |
| $p = 0.2$ | **65.1± 1.9%** | 61.1 ± 5.7% |
| $p = 0.25$ | **66.8 ± 3.3%** | 61.8 ± 6.5% |
| $p = 0.3$ | **65.1± 3.4%** | 52.8 ± 18.0% |

## E. More baseline comparisons

In this section, we will compare more baseline methods with the DomainBed settings. This will includes predicted group invariance (PGI,[2]) and representation self challenging (RSC, [12]). The results are shown in Table 6. From Table 6, we can observe that AIL achieves better or competitive results among methods including some recently published algorithms.

Table 6. Test accuracy of different methods on the Colored MNIST dataset.

| Methods | ERM | IRM [3] | VREx [16] | RSC [12] | PGI [2] | AIL(proposed) |
|---|---|---|---|---|---|---|
| $p = 0.15$ | 57.2 ± 3.4% | 68.9 ± 4.3% | 65.5 ± 3.3% | 53.6 ± 0.7% | 61.2 ± 5.2% | **74.3± 0.9%** |
| $p = 0.2$ | 39.5 ± 0.9% | 67.7 ± 6.1% | 58.4 ± 1.3% | 41.2 ± 0.7% | **70.1 ± 5.4%** | 65.1± 1.9% |
| $p = 0.25$ | 28.7 ± 0.5% | 58.5 ± 0.3% | 55.2 ± 4.0% | 26.3 ± 1.1% | 65.7 ± 0.5% | **66.8 ± 3.3%** |
| $p = 0.3$ | 24.2 ± 0.4% | 50.0 ± 0.4% | **65.5 ± 9.4%** | 25.2 ± 0.4% | 59.6 ± 1.3% | 65.1± 3.4% |

### E.1. Comparison with IRM on SST-2

We compared the performance of AIL and IRM on both the Colored MNIST dataset and the Punctuated SST-2 dataset. The testing methods are similar to the DOMAINBED suite proposed in [10]. We trained the model using random hyperparameter

choices, and performed model selection in 20 trials using a test domain validation set. As the models are trained for a fixed number of steps, this model selection process limits the number of queries to test domain data. We then obtain the mean accuracy and standard deviation by running multiple tests.

Additional results on the Punctuated SST-2 dataset are shown in Figure 8. Across different difficulty levels, AIL shows similar or better performance compared to IRM. It is worth noting that AIL behaves better and more stable when $p$ is higher, as the min-max formulation of AIL can update its environment partition to avoid spurious relations. It's quite unusual to see both methods achieve best accuracy under $p = 0.3$, and the results of IRM have strong variation. We think it is due to noise in the Punctuated SST-2 dataset. In our implementation, the dev set in the original train-test split SST-2 dataset was used as test set. After partitioned into 3 environments and assigned punctuation, the out-of-distribution test set contained only a small number of samples that it may lead to a noisy test result. More accurate comparison can be done in future work by constructing new natural language processing datasets.

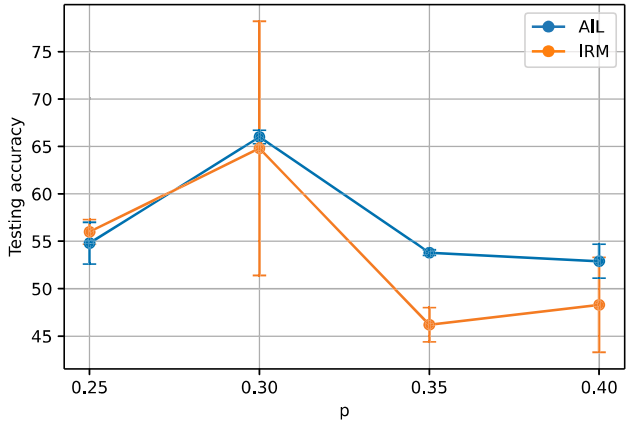


Figure 8. Comparison between AIL and IRM on the Punctuated SST-2 dataset.

## F. Additional experiment of AIL on PACS

Table 7. Classification accuracy on the PACS dataset with ResNet-18. The symbol * means the results are implemented by ourselves.

| Model | A | C | S | P | Average |
|---|---|---|---|---|---|
| ERM [3] | 77.85 | 74.86 | 67.74 | 95.73 | 79.05 |
| IRM [3]* | 70.31 | 73.12 | 75.51 | 84.73 | 75.92 |
| REx [16]* | 76.22 | 73.76 | 66.00 | 95.21 | 77.80 |
| JiGen [6] | 79.42 | 75.25 | 71.35 | **96.03** | 80.51 |
| CuMix [19] | **82.30** | 76.50 | 72.60 | 95.10 | 81.60 |
| MASF [9] | 80.29 | 77.17 | 71.69 | 94.99 | 81.03 |
| DRO [26]* | 78.09 | 74.18 | 77.00 | 93.45 | 80.68 |
| ***AIL*** | 82.28 | ***79.61*** | ***77.24*** | 95.21 | ***83.58*** |

PACS is commonly used in domain generalization [18]. It contains four domains with different image styles, namely photo, art painting, cartoon, and sketch. The variation in style introduces diversity shift across the training and the testing set that contain images from separate domains.

The implementation of AIL for PACS experiment follows [4], which consists of three main modules, a backbone network $f_{\boldsymbol{\theta}}(\cdot)$, a category branch $g_{\boldsymbol{\phi}}(\cdot)$, and a context branch $h_{\boldsymbol{\psi}}(\cdot)$ [4]. The two branches are simply two fully-connected layers without activation. Different from [4], we removed the semantic augmentation on context related feature and simply uses context feature to generate automatic environment splitting using clustering methods. Consider a training example $(\mathbf{x}, \mathbf{y})$, where $\mathbf{x}$ is the observation of data, the category branch is used to predict class label $\hat{\mathbf{y}}$ and the loss $\ell(\hat{\mathbf{y}}_1', \mathbf{y}_1)$ for this category branch is a cross-entropy loss. The context branch is used to automatic split environments $h_{\boldsymbol{\psi}}(t|\mathbf{x})$ using clustering methods.

The backbone network we chose is ResNet-18 initialized with pre-training weights on ImageNet. We used the leave-one-out method to calculate the testing accuracy for each target domain following [6]. We also conducted hyper-parameter

optimization (HPO) for proposed AIL algorithm. The average accuracy of these four environments is reported at the best performance with searched hyper-parameters.

AIL achieves ***state-of-the-art (SOTA)*** performance on PACS dataset. We will discuss this experiment in more detail in future studies.