# Antibody DomainBed: Towards robust predictions using invariant representations of biological sequences carrying complex distribution shifts

Anonymous Authors[1]

## Abstract

Recently, there has been an increased interest in accelerating drug design with machine learning (ML). Active ML-guided design of biological sequences with favorable properties involves multiple design cycles, in which (1) candidate sequences are proposed, (2) a subset of the candidates is selected using ML surrogate models trained to predict target properties of interest, and (3) a wet lab experimentally validates the selected sequences. The returned experimental results from one cycle provide valuable feedback for the next one, but the modifications they inspire in the candidate proposals or experimental protocol can lead to distribution shifts that impair the performance of surrogate models in the upcoming cycle. For the surrogate models to achieve consistent performance across cycles, we must explicitly account for the distribution shifts in their training. We turn to the notion of invariance and causal representation learning to achieve robustness across cycles. In particular, we apply domain generalization (DG) methods to develop invariant classifiers for predicting properties of therapeutic antibodies. We adapt a recent benchmark of DG algorithms, "DomainBed," to deploy 23 algorithms across 5 domains, or cycle numbers. Our results confirm that invariant features lead to better predictive performance for out-of-distribution domains.

## 1. Introduction

A model trained to minimize training error is incentivized to absorb all the correlations found in the training data. In many cases, however, the training data are not sampled independently from the same distribution as the test data and such a model may produce catastrophic failures outside the training domain [1, 2, 3, 4, 5]. The literature on domain generalization (DG) aims to build a robust predictor that will generalize to an unseen test domain. A popular approach in DG extracts a notion of domain **invariance** from datasets spanning multiple training domains [6, 7, 8]. This substantial body of work inspired by causality views the problem of DG as isolating the causal factors of variation, stable across domains, from spurious ones, which may change from training to test domains [8, 9, 10].

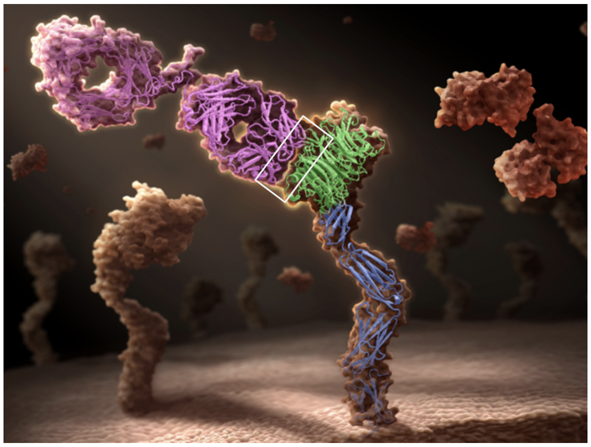


*Figure 1.* **Our prediction task: antibody-antigen binding.** Antibody Onartuzumab [2] (pink) binds to MET (green and blue), a lung cancer antigen target, on the cell surface. The strength of antibody-antigen binding is largely determined by the binding site of the antibody interacting with the antigen epitope, boxed in white.

Benchmarking efforts for DG algorithms, to date, have been largely limited to image classification tasks [e.g., 12, 13]. To prepare these algorithms for critical applications such as healthcare and medicine, we must validate and stress-test them on a wide variety of real-world datasets carrying selection biases, confounding factors, and other domain-specific idiosyncrasies. In this paper, we apply them to the problem of active drug design, a setting riddled with complex distribution shifts.

The specific application we consider is that of characterizing the **binding affinity** of therapeutic antibodies. Antibodies are proteins used by the immune system to recognize harmful foreign substances (antigens) such as bacteria and viruses [14]. They bind, or attach, to antigens in order to mediate an immune response against them. The strength of binding

[1] Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

is determined by the binding site of the antibody (paratope) interacting with the antigen epitope (Figure 1). Antibodies that bind tightly to a given target antigen are highly desirable as therapeutic candidates.

The wet-lab experiments that measure the binding affinity of antibodies are costly and time-consuming. In active antibody design, we thus assign a surrogate model to predict binding and select the most promising candidates for wet-lab evaluation based on the predictions. Developing an accurate surrogate model is a challenging task in itself, because, as explained in more detail in section 2, the model may latch onto **non-mechanistic** factors of variation in the data that do not cause binding: identity of the target antigen, assay used to measure binding, generative models (either human experts or ML) that proposed the antibody, and "batch effects" that create heteroscedastic measurement errors.

We approach active drug design from the DG perspective. Active drug design, executed in multiple design cycles, informs the DG algorithm development, as it abounds in distribution shifts previously underexplored in the DG literature. Conversely, it benefits from a robust (surrogate) binding predictor based on invariant representations. To summarize, the joint venture enables (1) impactful real-world benchmarking of DG algorithms and (2) development of robust predictors to serve active antibody design.

## 2. Accelerating antibody design with ML

**Problem formulation** Antibody design typically focuses on designing the variable region, which consists of two chains of amino acids. Each chain can be represented as a sequence of characters from an alphabet of 20 characters (for 20 possible amino acids). The entire variable region spans $L \sim 250$ amino acids on average. We denote the sequences as $\boldsymbol{x} = (a_1, \dots, a_L)$, where $a_l \in \{1, \dots, 20\}$ corresponds to the amino acid type at position $l \in [L]$. We experimentally measure the binding affinity $z \in \mathbb{R}$ from each sequence. But for simplicity, we create a binary classification task by creating a binary label $y \in \{0, 1\}$ from $z$. We set $y = 1$ if $z$ exceeds a chosen minimum affinity value that would qualify as binding and $y = 0$ otherwise. Each antibody $\boldsymbol{x}_i$, indexed $i$, carries a label $y_i$ in one of the design rounds $r$, where $r \in \{1, \dots, 5\}$. The labeled dataset for a round $r$ can thus be represented as a set of $n_r$ ordered pairs: $\mathcal{D}_r = \{(\boldsymbol{x}_i^r, y_i^r)\}_{i=1}^{n_r}$.

**Lab in the loop** Our antibody binding dataset is generated from an active ML-guided design process involving multiple design cycles, or rounds. As illustrated in Figure 2, each round consists of the following steps:

- **Step 1**. Millions of candidate sequences are sampled from a suite of generative models, including variational autoencoders [15, 16], energy-based models [17], and diffusion models [18, 19].
- **Step 2**. A small subset of several hundred promising candidates is selected based on binding predictions from a surrogate binding classifier.
- **Step 3**. The wet lab experimentally measures binding.
- **Step 4**. All models (generative and discriminative) are updated upon receiving new measurements.

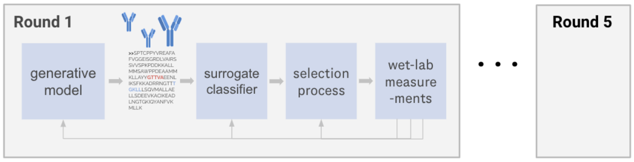


*Figure 2.* **Lab in the loop**, the active ML-guided antibody design process that generated our dataset.

In Step 4, both the generative model and the surrogate classifier $\hat{f}_\theta$ are updated. Beyond being refit on the new data returned from the lab, the generative models may undergo more fundamental modifications in their architectures, pretrained weights, and training/regularization schemes.

A standard approach to supervised learning tasks is empirical risk minimization (ERM) [20]. Let us first define the risk in each round $r$ as

$$\mathcal{R}^r(\theta) = \mathbb{E}_{(X^r, Y^r) \sim \mathcal{D}_{r_j}} \ell\left(\hat{f}_\theta(X), Y\right), \tag{1}$$

where $\ell$ is the loss function. ERM simply minimizes the training error, i.e., the average risk across all the training examples from all the rounds.

$$\begin{aligned} \mathcal{R}_{\text{ERM}}(\theta) &= \mathbb{E}_{(X^r, Y^r) \sim \bigcup_{j \in [5]} \mathcal{D}_{r_j}} \ell\left(\hat{f}_\theta(X), Y\right) \quad (2) \\ &= \mathbb{E}_{r \sim p_{\text{train}}(r)} \mathcal{R}^r(\theta), \quad (3) \end{aligned}$$

where $p_{\text{train}}(r)$ denotes distribution of the rounds in the training set. When we trained our surrogate classifier by ERM, it did not improve significantly even as the training set size increased over design rounds. In each subsequent round, representing the test domain, we observed that the classifier performance was close to random.

## 3. Domain generalization by invariance

The new measurements from the wet lab inspire modifications in the candidate proposals or experimental protocol, which lead to (feedback) covariate shift.

DG has recently gained traction in the ML community as concerns about productionalizing ML models in unseen test environments have emerged [21]. One line of research borrowing from Bayesian deep learning incorporates the predictor's uncertainty at test time [22]. Methods based on data augmentation apply either automated modifications to prevent overfitting [23] or counterfactual augmentations to enforce invariance between learned features [24, 25]. In this paper, we consider approaches inspired by invariant causal prediction (ICP) [26].

ICP frames DG in the language of causality and assumes that the data are generated according to a structural equation model (SEM) relating variables in a dataset to their parents by a set of mechanisms, or structural equations. The major assumption of ICP is the partitioning of the data into environments such that *each environment corresponds to interventions on the SEM*, but importantly, the mechanism by which the target variable is generated via its direct parents is unaffected [27]. This means that the true causal mechanism of the target variable is fixed, while other features of the generative distribution can vary. This motivates the objective of searching learning mechanisms that are stable (invariant) across environments with the hope that they would generalize under unseen, valid [3] interventions.

The ultimate goal of these frameworks is to attempt to learn an "optimal invariant predictor" which uses only the invariant features of the SEM. Similar to many tasks in ML, it is more convenient to build our methods in the manifold paradigm. That is, we assume that high-dimensional observations take lower-dimensional representations governed by a generative model. In the invariant learning paradigm, it is common to define the task as learning invariant representations of the data, rather than seeking invariant features in the observation space.

**Algorithms for invariant risk minimization** The goal of invariance-inspired DG methods is to learn representations that are invariant across interventions, or training environments. Formally, we follow:

**Definition 1** ([8]). *We say that a data representation $\Phi : X \to H$ elicits an invariant predictor $w \cdot \Phi$ across environments $E$ if there is a classifier $w : H \to Y$ simultaneously optimal for all environments, that is, $w \in \operatorname{argmin}_{\bar{w}:H\to Y} \mathcal{R}^e(\bar{w} \cdot \Phi)$ for all $e \in E$, where $\mathcal{R}^e(f) := \mathbb{E}_{(X^e,Y^e)}[\ell(f(X^e), Y^e)]$ (analogous to Equation 1).*

This problem setup has inspired a plethora of works, such as IRM [8]

$$\mathcal{R}_{\text{IRM}} = \min_{\substack{\Phi:\mathcal{X}\to\mathcal{H};\\ w:\mathcal{H}\to\mathcal{Y}}} \sum_{e\in E_{tr}} \mathcal{R}^e(w \cdot \Phi)$$

subject to $w \in \underset{\bar{w}:H\to Y}{\arg\min}\, \mathcal{R}^e(\bar{w} \cdot \Phi)$ for all $e \in E$. IRM assumes invariance of $\mathbb{E}[y|\Phi(x)]$—that is, invariance of the feature-conditioned label distribution. Follow-up studies make a stronger assumption on invariance based on higher-order conditional moments [28, 29]. Though this perspective has gained traction in the last few years, it is somewhat similar to the existing concepts of covariate shift, such as domain adaptation using meta learning. Thus, in our evaluation study we include invariance-inspired, but also domain adapration and meta-learning baselines.

**Hypothesis - invariant feature representations of antibodies** Our lab-in-the-loop (section 2) offers a unique testbed for DG algorithms. In particular, we attempt to answer the question:

*Can invariant representations help in developing robust predictors in the context of antibody design?*

We propose to consider the design rounds $r \in \{1, \ldots, 5\}$ as environments $e$, since rounds do correspond to valid interventions — our design cycles should not impact the true causal mechanism governing binding affinity. There are two types of features that a binding classifier can learn:

- *Invariant (causal) features*: various physico-chemical and geometric properties at the interface of antibody-antigen binding (Figure 1) and
- *Spurious correlations*: Other, round-specific features that are byproducts of different folding algorithms, generative models and their specific details, measurement assay types, etc.

We expect DG algorithms to be able to distinguish between the two, and only make use of the features invariant across rounds in their predictions.

## 4. Antibody DomainBed

Different DG solutions assume different types of invariance, and propose algorithms to estimate them from data. DomainBed [12] is a benchmark suite that contains the majority of DG algorithms developed in the past two years and a benchmark environment that compares them across multiple natural image datasets.

To adjust to our antibody design context, we modify DomainBed to accept biological sequences as input. We do so by (i) implementing a dataset loader for aligned antibody sequence representation and (ii) changing the ResNet [30] architecture to a more sequence-appropriate one, which includes positional encoding to take into account the ordering of amino acids in a biological sequence. Figure 4 depicts our framework. As antibody-antigen binding depends on the interface between the two proteins, we need to account for the various possible antigen targets. We thus include the antigen sequence in the input to the classifier, by concatenating the antibody sequence with the antigen sequence.

From the available DG algorithms in DomainBed, we evaluate 23 baselines with 10 hyperparameter configurations (with varying batch size, weight decay, and learning rate) and 3 seed repetitions for each configuration. That yields a

[3] Interventions are considered valid if they do not change the structural equation of $Y$.

| Algorithm | Round 1 | Round 2 | Round 3 | Round 4 | Round 5 | Avg |
|---|---|---|---|---|---|---|
| **ERM** | 90.4 ± 1.8 | 78.0 ± 0.2 | 72.6 ± 1.7 | 69.4 ± 3.0 | 65.4 ± 1.8 | 75.2 |
| **Fish** | 96.8 ± 0.2 | **79.2 ± 0.5** | 72.1 ± 0.6 | 64.9 ± 0.9 | 69.5 ± 1.0 | **76.5** |
| **IRM** | 93.2 ± 1.6 | 77.5 ± 2.1 | 74.2 ± 1.0 | 63.0 ± 0.2 | 68.7 ± 1.5 | 75.3 |
| **GroupDRO** | 93.2 ± 0.7 | 71.6 ± 0.3 | 72.6 ± 1.1 | 71.9 ± 3.9 | 59.5 ± 1.2 | 73.8 |
| **Mixup** | 94.1 ± 1.9 | 77.7 ± 1.0 | 73.8 ± 3.1 | 68.4 ± 1.0 | 63.4 ± 2.3 | 75.5 |
| **CORAL** | 91.3 ± 2.3 | 76.2 ± 1.4 | 72.1 ± 1.8 | 68.0 ± 0.9 | 66.7 ± 0.9 | 74.9 |
| **MMD** | 86.1 ± 1.1 | 72.8 ± 0.6 | 71.3 ± 0.2 | 68.7 ± 2.2 | 61.3 ± 0.3 | 72.0 |
| **DANN** | 93.6 ± 2.5 | 72.3 ± 0.2 | 69.2 ± 2.6 | 53.8 ± 2.9 | 68.0 ± 1.3 | 71.4 |
| **MTL** | 93.1 ± 1.9 | 76.3 ± 1.1 | 69.9 ± 0.6 | 68.4 ± 0.2 | 65.4 ± 2.7 | 74.6 |
| **SagNet** | 93.3 ± 2.6 | 76.9 ± 1.8 | 69.7 ± 2.3 | 71.4 ± 2.4 | 67.6 ± 0.7 | 75.8 |
| **VREx** | 94.6 ± 0.9 | 77.7 ± 1.0 | 68.9 ± 2.6 | 68.4 ± 2.1 | 67.3 ± 0.4 | 75.4 |
| **SD** | 92.8 ± 1.8 | 77.8 ± 0.3 | **75.4 ± 1.6** | **74.3 ± 1.6** | 62.2 ± 2.0 | **76.5** |
| **ANDMask** | 88.4 ± 5.6 | 77.7 ± 2.2 | 58.8 ± 5.7 | 61.1 ± 3.3 | 73.5 ± 2.7 | 71.9 |
| **SANDMask** | 90.9 ± 1.2 | 76.6 ± 1.6 | 70.8 ± 0.4 | 70.6 ± 1.0 | 66.7 ± 1.8 | 75.1 |
| **IGA** | **98.4 ± 1.3** | 78.9 ± 0.7 | 65.6 ± 0.4 | 59.0 ± 1.4 | 66.5 ± 4.6 | 73.7 |
| **Fishr** | 92.7 ± 2.1 | 76.9 ± 0.5 | 74.2 ± 0.3 | 69.3 ± 0.4 | 68.4 ± 0.9 | 76.3 |
| **TRM** | 93.1 ± 1.3 | 77.1 ± 0.7 | 72.5 ± 1.0 | 71.4 ± 1.6 | 66.3 ± 0.5 | 76.0 |
| **IB ERM** | 90.0 ± 1.7 | 77.4 ± 0.1 | 73.0 ± 0.7 | 68.5 ± 1.5 | 66.5 ± 1.4 | 75.1 |
| **IB IRM** | 96.7 ± 0.9 | 78.6 ± 0.9 | 65.0 ± 7.1 | 63.1 ± 0.2 | 71.5 ± 0.9 | 75.0 |
| **Transfer** | 98.2 ± 1.3 | 76.9 ± 2.1 | 54.9 ± 5.0 | 53.7 ± 2.6 | **74.1 ± 1.9** | 71.6 |
| **CausIRL CORAL** | 92.2 ± 3.2 | 75.0 ± 1.2 | 72.8 ± 1.2 | 70.1 ± 0.3 | 66.5 ± 2.2 | 75.3 |
| **CausIRL MMD** | 91.6 ± 2.4 | 74.1 ± 1.6 | 74.2 ± 0.8 | 71.5 ± 4.1 | 63.9 ± 3.1 | 75.1 |
| **EQRM** | 93.8 ± 1.4 | 76.5 ± 1.2 | 71.8 ± 0.7 | 67.9 ± 1.2 | 66.8 ± 0.2 | 75.4 |
| Average per round | 93.0 | 76.5 | 70.2 | 66.8 | 66.8 | 74.7 |

*Table 1.* Accuracy. Higher is better. Error bars are across three seed repetitions. Algorithms outperforming (underperforming) ERM are highlighted in green (red).

| Algorithm | Round 1 | Round 2 | Round 3 | Round 4 | Round 5 | Avg |
|---|---|---|---|---|---|---|
| **ERM** | 21.7 ± 2.9 | 56.1 ± 3.0 | 62.8 ± 4.4 | 58.3 ± 2.5 | 65.5 ± 5.3 | 52.9 |
| **Fish** | 23.0 ± 2.4 | **47.4 ± 1.1** | 59.6 ± 1.8 | 62.5 ± 4.6 | 56.7 ± 3.3 | **49.8** |
| **IRM** | 25.5 ± 2.6 | 52.2 ± 0.4 | 63.0 ± 2.7 | 67.2 ± 1.0 | 55.9 ± 1.6 | 52.8 |
| **GroupDRO** | 19.8 ± 2.4 | 56.6 ± 3.1 | 58.6 ± 3.5 | 64.4 ± 3.8 | 60.8 ± 3.0 | 52.0 |
| **Mixup** | 27.8 ± 1.5 | 49.3 ± 0.6 | 62.8 ± 2.8 | 69.2 ± 4.8 | 60.5 ± 2.9 | 53.9 |
| **CORAL** | 24.2 ± 3.6 | 54.4 ± 3.5 | 59.9 ± 3.7 | 64.0 ± 4.3 | 69.1 ± 3.1 | 54.3 |
| **MMD** | 41.7 ± 4.5 | 48.8 ± 0.2 | 61.3 ± 3.9 | 66.2 ± 3.3 | 52.2 ± 1.6 | 54.1 |
| **DANN** | 22.6 ± 0.9 | 54.7 ± 0.5 | 71.7 ± 6.6 | 87.6 ± 8.3 | 60.3 ± 3.0 | 59.4 |
| **MTL** | 24.0 ± 3.6 | 61.2 ± 5.6 | 66.7 ± 3.7 | 58.1 ± 0.6 | 62.4 ± 2.6 | 54.5 |
| **SagNet** | 16.3 ± 1.5 | 52.8 ± 1.3 | 61.5 ± 2.6 | 57.1 ± 5.0 | 61.4 ± 4.7 | **49.8** |
| **VREx** | 30.1 ± 1.6 | 49.1 ± 0.2 | 64.4 ± 1.4 | 63.2 ± 3.5 | **51.6 ± 2.0** | 51.7 |
| **SD** | 16.6 ± 2.2 | 55.1 ± 3.2 | 62.4 ± 1.0 | 55.6 ± 3.1 | 61.5 ± 1.8 | 50.3 |
| **ANDMask** | 17.6 ± 2.6 | 52.7 ± 2.3 | 72.2 ± 4.6 | 69.3 ± 1.3 | 64.9 ± 11.5 | 55.3 |
| **SANDMask** | 20.5 ± 4.2 | 66.1 ± 7.9 | 58.4 ± 2.3 | 70.4 ± 4.2 | 55.9 ± 5.0 | 54.3 |
| **IGA** | 42.0 ± 2.3 | 49.5 ± 0.8 | 83.2 ± 4.2 | 74.4 ± 4.3 | 58.2 ± 2.2 | 61.4 |
| **Fishr** | 21.2 ± 2.1 | 59.7 ± 5.0 | 58.6 ± 2.5 | **55.5 ± 1.4** | 57.1 ± 2.7 | 50.4 |
| **TRM** | 26.7 ± 3.5 | 52.6 ± 3.9 | **54.9 ± 0.8** | 63.3 ± 2.8 | 62.8 ± 1.3 | 52.1 |
| **IB ERM** | 21.3 ± 2.5 | 54.2 ± 2.6 | 63.2 ± 0.8 | 60.8 ± 4.2 | 59.2 ± 1.5 | 51.8 |
| **IB IRM** | 30.0 ± 6.0 | 50.0 ± 0.2 | 69.7 ± 0.5 | 70.9 ± 1.2 | 57.2 ± 0.9 | 55.6 |
| **Transfer** | 25.7 ± 6.9 | 53.9 ± 2.6 | 79.0 ± 8.6 | 69.3 ± 0.5 | 54.8 ± 1.8 | 56.5 |
| **CausIRL CORAL** | 22.6 ± 2.7 | 55.7 ± 3.9 | 60.6 ± 1.9 | 61.4 ± 1.9 | 62.1 ± 4.6 | 52.5 |
| **CausIRL MMD** | 24.6 ± 3.5 | 49.6 ± 1.3 | 63.7 ± 2.8 | 68.1 ± 3.1 | 61.1 ± 3.1 | 53.4 |
| **EQRM** | **13.9 ± 2.7** | 59.7 ± 7.9 | 60.5 ± 2.0 | 63.7 ± 1.8 | 61.2 ± 4.8 | 51.8 |
| Average per round | 24.3 | 54.0 | 64.3 | 65.2 | 59.7 | 53.5 |

*Table 2.* Negative log likelihood. Lower is better. Error bars are across three seed repetitions. Algorithms outperforming (underperforming) ERM are highlighted in green (red).

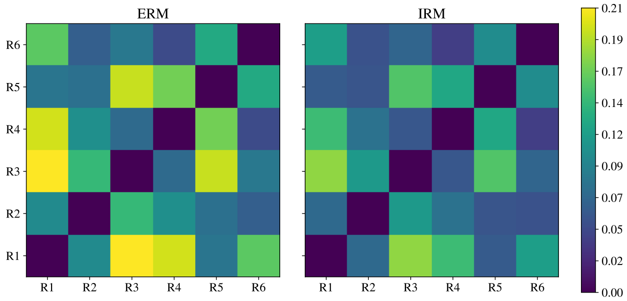


*Figure 3.* MMD (cosine kernel) in the learned representation between every pair of rounds. The latent feature space of IRM is more uniform across rounds than that of ERM, as expected.

total of 3600 experiments. We report the results from *model selection method: training domain validation set*, which is a leave-one-environment-out model selection strategy. We (1) split the data into train and test environments, (2) pool the validation sets of each training domain to create an overall validation set, and (3) choose the model maximizing the accuracy on the pooled validation set.

We open-source our efforts so that other researchers can continue further evaluations on similar biological datasets. With this paper, we make publicly available the "Antibody DomainBed," a codebase aligned with the DomainBed suite, at anonymous-link. We also plan to release a public benchmark antibody dataset available at anonymous-link.

We test the following algorithms: ERM [31], Fish [32], IRM [8], GroupDRO [33], Mixup [34], CORAL [35], MMD [36], DANN [37], CDANN [38], MTL [39], SagNet [40], VREx [28], SD [41], ANDMask [42], SANDMask [43], IGA [44], Fishr [10], TRM [45], IB-ERM and IB-IRM [9], Transfer [46], CausIRL CORAL and CausIRL MMD [47], and EQRM [48]. Appendix A gives a brief description of each baseline. See the references for more details.

## 5. Summary and Outlook

We applied DG algorithms to the problem of developing an antibody binding classifier robust to non-mechanistic features of the design rounds. Table 1 and Table 2 present the accuracy and negative log-likelihood, respectively, for the chosen model across the three seeds. Similarly to the conclusions from DomainBed on images, when model selection is done over a large grid of hyperparameters, it is difficult to conclude if there is consistent improvement when leveraging invariant feature representations. In each round, however, there are at least a few DG algorithms that achieve better results than ERM. Moreover, domain adaptation algorithms do not outperform ERM, while most invariance-inspired algorithms do, especially in the later rounds. While performance varies across algorithms, on the whole, (1) earlier rounds seem to be easier environments for all baselines and (2) invariant features appear to help (for each round there is always at least one IRM-variant that does better) with rounds expected to have the greatest distribution shifts, that is, the later rounds 3-5. We also examine the MMD distance in the learned representations between the rounds for ERM and IRM Figure 3, averaged over multiple runs. IRM embeddings are more similar between rounds compared to ERM, and can be viewed as more stable representations of the antibodies across rounds.

Encouraged by these results, we are (1) working on their deployment in the next round of our active drug design and (2) open-sourcing a distribution shift benchmark focused on biological sequences to motivate other ML researchers to target impactful real-world applications closer to the production setting.

## References

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

[2] John R Zech, Marcus A Badgeley, Manway Liu, Anthony B Costa, Joseph J Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. *PLoS medicine*, 15(11):e1002683, 2018.

[3] Sara Beery, Dan Morris, and Siyu Yang. Efficient pipeline for camera trap image review. *arXiv preprint arXiv:1907.06772*, 2019.

[4] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.

[5] Yannic Neuhaus, Maximilian Augustin, Valentyn Boreiko, and Matthias Hein. Spurious features everywhere–large-scale detection of harmful spurious features in imagenet. *arXiv preprint arXiv:2212.04871*, 2022.

[6] Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. *Advances in neural information processing systems*, 24, 2011.

[7] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International conference on machine learning*, pages 10–18. PMLR, 2013.

[8] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

[9] Kartik Ahuja, Ethan Caballero, Dinghuai Zhang, Jean-Christophe Gagnon-Audet, Yoshua Bengio, Ioannis Mitliagkas, and Irina Rish. Invariance principle meets information bottleneck for out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 34:3438–3450, 2021.

[10] Alexandre Rame, Corentin Dancette, and Matthieu Cord. Fishr: Invariant gradient variances for out-of-distribution generalization. In *International Conference on Machine Learning*, pages 18347–18377. PMLR, 2022.

[11] Christian Rolfo, Nele Van Der Steen, Patrick Pauwels, and Federico Cappuzzo. Onartuzumab in lung cancer: the fall of icarus?, 2015.

[12] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

[13] Aengus Lynch, Gbètondji JS Dovonon, Jean Kaddour, and Ricardo Silva. Spawrious: A benchmark for fine control of spurious correlation biases. *arXiv preprint arXiv:2303.05470*, 2023.

[14] Surjit Singh, Nitish K Tank, Pradeep Dwiwedi, Jaykaran Charan, Rimplejeet Kaur, Preeti Sidhu, and Vinay K Chugh. Monoclonal antibodies: a review. *Current clinical pharmacology*, 13(2):85–99, 2018.

[15] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[16] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

[17] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. *Advances in Neural Information Processing Systems*, 32, 2019.

[18] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[20] VN Vapnik. Principles of risk minimization for learning theory, advances in neural information processing nips 4 (pp. 831±838), 1992.

[21] Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. The risks of invariant risk minimization. *ICLR*, 2021.

[22] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[23] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[24] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. *Advances in neural information processing systems*, 31, 2018.

[25] Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. Learning the difference that makes a difference with counterfactually-augmented data. *arXiv preprint arXiv:1909.12434*, 2019.

[26] Jonas Peters, Peter Bühlmann, and Nicolai Meinshausen. Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, pages 947–1012, 2016.

[27] Judea Pearl. *Causality*. Cambridge university press, 2009.

[28] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex). In *International Conference on Machine Learning*, pages 5815–5826. PMLR, 2021.

[29] Chuanlong Xie, Fei Chen, Yue Liu, and Zhenguo Li. Risk variance penalization: From distributional robustness to causality. *arXiv preprint arXiv:2006.07544*, 1, 2020.

[30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[31] Vladimir Vapnik. Statistical learning theory. *(No Title)*, 1998.

[32] 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.

[33] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

[34] Shen Yan, Huan Song, Nanxiang Li, Lincan Zou, and Liu Ren. Improve unsupervised domain adaptation with mixup training. *arXiv preprint arXiv:2001.00677*, 2020.

[35] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14*, pages 443–450. Springer, 2016.

[36] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5400–5409, 2018.

[37] 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.

[38] 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.

[39] Gilles Blanchard, Aniket Anand Deshmukh, Ürun Dogan, Gyemin Lee, and Clayton Scott. Domain generalization by marginal transfer learning. *The Journal of Machine Learning Research*, 22(1):46–100, 2021.

[40] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap by reducing style bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8690–8699, 2021.

[41] Mohammad Pezeshki, Oumar Kaba, Yoshua Bengio, Aaron C Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. *Advances in Neural Information Processing Systems*, 34:1256–1272, 2021.

[42] Giambattista Parascandolo, Alexander Neitz, Antonio Orvieto, Luigi Gresele, and Bernhard Schölkopf. Learning explanations that are hard to vary. *arXiv preprint arXiv:2009.00329*, 2020.

[43] Soroosh Shahtalebi, Jean-Christophe Gagnon-Audet, Touraj Laleh, Mojtaba Faramarzi, Kartik Ahuja, and Irina Rish. Sand-mask: An enhanced gradient masking strategy for the discovery of invariances in domain generalization. *arXiv preprint arXiv:2106.02266*, 2021.

[44] Masanori Koyama and Shoichiro Yamaguchi. When is invariance useful in an out-of-distribution generalization problem? *arXiv preprint arXiv:2008.01883*, 2020.

[45] Yilun Xu and Tommi Jaakkola. Learning representations that support robust transfer of predictors. *arXiv preprint arXiv:2110.09940*, 2021.

[46] Guojun Zhang, Han Zhao, Yaoliang Yu, and Pascal Poupart. Quantifying and improving transferability in domain generalization. *Advances in Neural Information Processing Systems*, 34:10957–10970, 2021.

[47] Mathieu Chevalley, Charlotte Bunne, Andreas Krause, and Stefan Bauer. Invariant causal mechanisms through distribution matching. *arXiv preprint arXiv:2206.11646*, 2022.

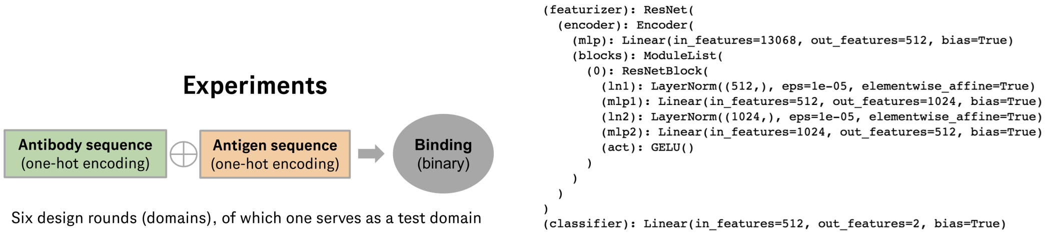
[48] Cian Eastwood, Alexander Robey, Shashank Singh, Julius Von Kügelgen, Hamed Hassani, George J Pappas, and Bernhard Schölkopf. Probable domain generalization via quantile risk minimization. *arXiv preprint arXiv:2207.09944*, 2022.

## A. DomainBed baselines

We briefly summarize the baselines evaluated in section 4:

- **Empirical Risk Minimization ERM**
- **Group Distributionally Robust Optimization (DRO])** ERM with increased importance of domains with larger errors.
- **Inter-domain Mixup** - performs ERM on linear interpolations of examples from random pairs of domains and their labels.
- **Marginal Transfer Learning (MTL)** from the perspective of information about test task being drawn from that task's marginal feature distribution
- **Meta-Learning for Domain Generalization (MLDG)** leverages MAML to meta-learn how to generalize across domains.
- **Spectral Decoupling (SD)** a regularization method that addressed Gradient Starvation which arises when cross-entropy loss is minimized by capturing only a subset of features relevant for the task, despite the presence of other predictive features that fail to be discovered.
- Different variants of the popular algorithm of Ganin et al. [2016] to learn features $\Phi(X_r)$ with distributions matching across domains:
  - **Domain-Adversarial Neural Networks (DANN)** employ an adversarial network to match feature distributions.
  - **Class-conditional DANN (CDAAN)** is a variant of DANN matching the conditional distributions $P(\Phi(X_r)|Y_r = y)$ across domains, for all labels y.
  - **CORAL** matches the mean and covariance of feature distributions.
  - **MMD** matches the mean maximum discrepancy of feature distributions.
- **Invariant Risk Minimization (IRM)** learns a feature representation such that the optimal linear classifier on top of that representation matches across domains.
- **Variance Risk Extrapolation (VAREx)** optimization over a perturbation set of extrapolated domains with a penalty on the variance of training risks.
- **ANDMask** trade convergence speed for invariance, by replacing the gradient descent average mean (logical OR) by geometric arithmetic mean between gradients logical AND.
- **Smoothed-AND mask** (SAND-mask) matching the Hessians of different environments.
- **Fish** an inter-domain gradient matching objective by maximizing the inner product between means of gradient distributions from different domains.
- **Fishr** match the domain level gradient variances, i.e., the second moment of the gradient distributions.
- **TRM** uses the per-environment optimal predictor to guide the representation learning.
- **IB-ERM** and **IB-IRM** adding an information bottleneck constraint along with invariance in the objective.
- Transfer optimising for the therein defined transferability metric, implemented through adversarial training/minimax optimization.
- **CausCORAL** and **CausMMD** - instead of taking pairwise distances across domains, they compute distances between batches that follow different domain distributions.

## B. Model architecture



```
(featurizer): ResNet(
  (encoder): Encoder(
    (mlp): Linear(in_features=13068, out_features=512, bias=True)
    (blocks): ModuleList(
      (0): ResNetBlock(
        (ln1): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
        (mlp1): Linear(in_features=512, out_features=1024, bias=True)
        (ln2): LayerNorm((1024,), eps=1e-05, elementwise_affine=True)
        (mlp2): Linear(in_features=1024, out_features=512, bias=True)
        (act): GELU()
      )
    )
  )
)
(classifier): Linear(in_features=512, out_features=2, bias=True)
```

*Figure 4.* Details on model architecture, featurizer, and linear classifier.