SUPERVISED CONTRASTIVE BLOCK DISENTANGLEMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Real-world datasets often combine data collected under different experimental conditions. Although this yields larger datasets, it also introduces spurious correlations that make it difficult to accurately model the phenomena of interest. We address this by learning two blocks of latent variables to independently represent the phenomena of interest and the spurious correlations. The former are correlated with the target variable y and invariant to the environment variable e, while the latter depend on e. The invariance of the phenomena of interest to e is highly sought-after but difficult to achieve on real-world datasets. Our primary contribution is an algorithm called Supervised Contrastive Block Disentanglement (SCBD) that is highly effective at enforcing this invariance. It is based purely on supervised contrastive learning, and scales to real-world data better than existing approaches. We empirically validate SCBD on two challenging problems. The first is domain generalization, where we achieve strong performance on a synthetic dataset, as well as on Camelyon17-WILDS. SCBD introduces a single hyperparameter α that controls the degree of invariance to e. When we increase α to strengthen the degree of invariance, there is a monotonic improvement in out-of-distribution performance at the expense of in-distribution performance. The second is a scientific problem of batch correction. Here, we demonstrate the utility of SCBD by learning representations of single-cell perturbations from 26 million Optical Pooled Screening images that are nearly free of technical artifacts induced by the variation across wells.

1 INTRODUCTION

Real-world machine learning (ML) datasets often combine data collected under different experimental conditions, such as medical images or stained histopathology sections collected at different hospitals (Bándi et al., 2019; McKinney et al., 2020). This practice yields larger datasets, but the different experimental conditions alter the images' appearance, and induce spurious correlations that make it difficult to model the phenomena of interest. While human perception is relatively robust (Makino et al., 2022b), ML models tend to rely on hospital-specific spurious correlations, and fail to generalize out-of-distribution to unseen hospitals (Koh et al., 2021).

Similar spurious correlations are a long-standing problem in experimental biology (Chandrasekaran et al., 2024), where they are called batch effects (Leek et al., 2010). They can arise between experiments conducted in different labs, within the same lab, and even within a single large parallelized experiment. Removing batch effects by batch correction is a highly-active research direction (Arevalo et al., 2024). Batch effects have been ubiquitously observed across various high-throughput lab techniques. For example, consider image-based perturbation screening, where the goal is to understand the effect of perturbations by comparing the images of perturbed cells to those from a control condition. When the images also vary due to differences in experimental conditions, this confounds our understanding of the effect of the perturbations.

In some cases, we have prior knowledge of the spurious correlations, and can take steps to remove them. For example, color-based data augmentation can remove the staining variation in histopathology images, yielding significant improvements in out-of-distribution generalization (Nguyen et al., 2023). Similarly, in experimental biology, there are manually-engineered post-processing methods that remove specific known batch effects (Carpenter et al., 2006). However, such manual approaches

have two important limitations. First, they require manual post-hoc quality checks to ensure the post-processing did not remove variations of interest. Second, some spurious correlations may be unknown, and therefore uncorrected. This motivates the development of automated approaches that maximize the removal of the spurious correlations, while minimizing the impact on the phenomena of interest.

To address these issues, we propose to learn representations of the data using two blocks of latent variables, one encoding the variation of interest, and the other encoding the spurious correlations. We break symmetry between the blocks by exploiting the supervisory signal of the target variable y and the environment variable e. Let $\mathbf{x}$ be the observation, such as a histopathology image. Let y represent the phenomenon of interest, such as the presence of disease. Finally, let e represent the experimental conditions, such as the hospital that processed the image. From these observed variables, we learn two embeddings $\mathbf{z}_c$ and $\mathbf{z}_s$, where $\mathbf{z}_c$ represents the variation of $\mathbf{x}$ induced by y, and $\mathbf{z}_s$ represents the variation of $\mathbf{x}$ induced by e. As we discuss in Section 2, we can also let $\mathbf{z}_s$ represent the variation of $\mathbf{x}$ induced by both y and e. Our goal is to *block disentangle* $\mathbf{z}_c$ and $\mathbf{z}_s$ so that they independently represent distinct information.

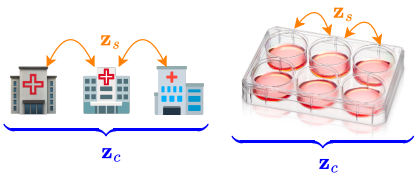


Figure 1: Spurious correlations emerge when collecting medical images from different hospitals, or conducting single-cell perturbation screens across multiple wells. $\mathbf{z}_s$ models these spurious correlations, while $\mathbf{z}_c$ models the environment-invariant correlations.

The promise of estimating $\mathbf{z}_c$ such as it represents the variation of $\mathbf{x}$ due to y in way that is invariant to e is significant for many high-impact downstream applications. However, existing methods for this task require additional regularization and hyperparameter tuning to ensure that $\mathbf{z}_c$ remains invariant with respect to e. Optimizing those hyperparameters in the presence of distribution shifts has proven challenging in practice (Gulrajani & Lopez-Paz, 2021). While a few existing approaches have shown success in simplified settings (Peters et al., 2016; Ganin et al., 2016; Louizos et al., 2016; Lopez et al., 2018; Arjovsky et al., 2019; Lu et al., 2021; Kong et al., 2022), most methods tested on real-world data have not outperformed carefully-tuned supervised learning baselines (Gulrajani & Lopez-Paz, 2021). Consequently, the problem of learning block-disentangled representations remains largely unsolved.

Our primary contribution is an algorithm called Supervised Contrastive Block Disentanglement (SCBD). We claim that SCBD achieves the desired invariance to e with minimal and interpretable hyperparameter tuning. Unlike prior work on block disentanglement that use variational or adversarial objectives, our algorithm is based purely on Supervised Contrastive Learning (SCL) (Khosla et al., 2020). Following the authors' notation, we learn two encoder networks $\mathrm{Enc}_c(\cdot)$ and $\mathrm{Enc}_s(\cdot)$ that map $\mathbf{x}$ to the representations given by

$$\mathbf{r}_c \coloneqq \mathrm{Enc}_c(\mathbf{x}) \in \mathbb{R}^{D_{\mathbf{r}_c}}, \quad \mathbf{r}_s \coloneqq \mathrm{Enc}_s(\mathbf{x}) \in \mathbb{R}^{D_{\mathbf{r}_s}}.$$

We additionally learn two projection networks $\mathrm{Proj}_c(\cdot)$ and $\mathrm{Proj}_s(\cdot)$ that map the representations to the lower-dimensional embeddings given by

$$\mathbf{z}_c \coloneqq \mathrm{Proj}_c(\mathbf{r}_c) \in \mathbb{R}^{D_{\mathbf{z}_c}}, \quad \mathbf{z}_s \coloneqq \mathrm{Proj}_s(\mathbf{r}_s) \in \mathbb{R}^{D_{\mathbf{z}_s}},$$

which are normalized to the unit hypersphere. Finally, we learn a decoder $\mathrm{Dec}(\mathbf{z}_c, \mathbf{z}_s)$ which reconstructs $\mathbf{x}$ from $\mathbf{z}_c$ and $\mathbf{z}_s$. The optimization objective consists of four losses, and is given by

$$\min \mathcal{L}^{\text{sup}}_{\mathbf{z}_c,y} + \mathcal{L}^{\text{sup}}_{\mathbf{z}_s,e} + \alpha \mathcal{L}^{\text{inv}}_{\mathbf{z}_c,e} - \log p(\mathbf{x} \mid \mathrm{Dec}(\mathbf{z}_c, \mathbf{z}_s)). \tag{1}$$

The first loss directly applies SCL to cluster $\mathbf{z}_c$ with respect to y. Similarly, the second loss directly applies SCL to cluster $\mathbf{z}_s$ with respect to e. The third loss is our novel invariance loss, which is also based on SCL, and ensures that $\mathbf{z}_c$ is well-mixed with respect to e. In other words, our invariance loss purges $\mathbf{z}_c$ of the influence of e. The fourth loss is a reconstruction loss. We describe these losses in detail in Section 2. SCBD incorporates a single hyperparameter $\alpha \in \mathbb{R}_{\geq 0}$ to adjust the degree to which $\mathbf{z}_c$ is invariant to e. When we increase α, we observe a monotonic improvement on several downstream evaluation metrics that benefit from block disentanglement.

We empirically validate SCBD on three datasets spanning two challenging real-world problems. The first problem is domain generalization (Blanchard et al., 2011; Muandet et al., 2013), where

$\mathbf{z}_c$ represents features whose correlation with y is invariant to e. We use SCBD to generalize out-of-distribution on the synthetic Colored MNIST (CMNIST) dataset, as well as the real-world histopathology dataset Camelyon17-WILDS (Koh et al., 2021). We demonstrate that SCBD enables precise control over the trade-off between in-distribution and out-of-distribution generalization performance through adjustment of the hyperparameter α. Additionally, we show that on both datasets, SCBD achieves better out-of-distribution performance relative to the conventional baselines in the literature.

The second problem is batch correction, where we apply SCBD to a dataset of images of over 26 million individual cells (Funk et al., 2022). The cells are treated with $5{,}050$ genetic perturbations which are labeled as y, and collected across 34 wells which are labeled as e. We use SCBD to represent the effect of the perturbation with $\mathbf{z}_c$, and the variation across wells with $\mathbf{z}_s$. We show that relative to strong baselines, SCBD provides estimates of $\mathbf{z}_c$ that preserves biological signal and are significantly less sensitive to batch effects.

2 Supervised Contrastive Block Disentanglement

We now define the individual terms in the SCBD optimization objective in Equation 1. Our starting point is a probabilistic interpretation of SCL that helps derive our novel invariance loss. Following the notation from Khosla et al. (2020), let I be the set of indices of examples within a minibatch. For each anchor point $i \in I$, we denote the set of the remaining examples as $\mathcal{A}(i) = I \setminus \{i\}$. In SCL, anchor points are compared to other examples via their dot product. We define $|\mathcal{A}(i)|$ independent Bernoulli random variables $M^j_{i,c}$ for j in $\{1, \ldots, |\mathcal{A}(i)|\}$ to represent whether $\mathbf{z}^i_c$ is matched with $\mathbf{z}^j_c$. The matching probability is defined as

$$P(M^j_{i,c} = 1) = \frac{\exp(\mathbf{z}^i_c \cdot \mathbf{z}^j_c / \tau)}{\sum_{a \in \mathcal{A}(i)} \exp(\mathbf{z}^i_c \cdot \mathbf{z}^a_c / \tau)}.$$

The notion of matching is relative to the rest of the examples in $\mathcal{A}(i)$, which is why we use a softmax despite the random variable being binary. A similar definition holds for the random variable $M^j_{i,s}$, defined in the space of $\mathbf{z}_s$.

The first term in Equation 1 is a direct application of SCL, and is given by

$$\mathcal{L}^{\text{sup}}_{\mathbf{z}_c, y} = -\sum_{i \in I} \frac{1}{|\mathcal{P}_y(i)|} \sum_{p \in \mathcal{P}_y(i)} \log P(M^p_{i,c} = 1),$$

where $\mathcal{P}_y(i) = \{j \in \mathcal{A}(i) : y^i = y^j\}$ are the positive pairs for the anchor point i with respect to y. This represents the negative log joint probability of observing the positive pairs, normalized by the number of positive pairs, and summed across all anchor points. Minimizing this loss clusters $\mathbf{z}_c$ with respect to y.

The second term in Equation 1 is also a direct application of SCL, and is given by

$$\mathcal{L}^{\text{sup}}_{\mathbf{z}_s, e} = -\sum_{i \in I} \frac{1}{|\mathcal{P}_e(i)|} \sum_{p \in \mathcal{P}_e(i)} \log P(M^p_{i,s} = 1),$$

where $\mathcal{P}_e(i) = \{j \in \mathcal{A}(i) : e^i = e^j\}$ are the positive pairs for the anchor point i with respect to e. Minimizing this loss clusters $\mathbf{z}_s$ with respect to e. As we later discuss in our experiments (Section 4.1), it can be useful to let $\mathbf{z}_s$ represent the variation of $\mathbf{x}$ with respect to the pair (y, e), rather than just e. We can do this by replacing $\mathcal{L}^{\text{sup}}_{\mathbf{z}_s, e}$ with

$$\mathcal{L}^{\text{sup}}_{\mathbf{z}_s, (y,e)} = -\sum_{i \in I} \frac{1}{|\mathcal{P}_{(y,e)}(i)|} \sum_{p \in \mathcal{P}_{(y,e)}(i)} \log P(M^p_{i,s} = 1),$$

where $\mathcal{P}_{(y,e)}(i) = \{j \in \mathcal{A}(i) : y^i = y^j, e^i = e^j\}$ are the positive pairs for the anchor point i with respect to the pairs of labels (y, e).

The third term in Equation 1 is our novel invariance loss. We define $\mathcal{N}_e(i) = \mathcal{A}(i) \setminus \mathcal{P}_e(i)$ as the negative pairs with respect to the label e. Since $\{\mathcal{P}_e(i), \mathcal{N}_e(i)\}$ is a partition of $\mathcal{A}(i)$, we consider

the binary classification task of whether $\mathbf{z}_c^i$ is more likely to be matched with its positive or negative pairs with respect to e. One way to perform this classification is to predict that i is more likely to be matched with $\mathcal{P}_e(i)$ if

$$\sum_{p \in \mathcal{P}_e(i)} \log P(M_{i,c}^p = 1) > \sum_{n \in \mathcal{N}_e(i)} \log P(M_{i,c}^n = 1).$$

Since our goal is to make $\mathbf{z}_c$ invariant to e, we optimize $\mathbf{z}_c$ to make this classifier fail. We do this by minimizing

$$\mathcal{L}_{\mathbf{z}_c,e}^{\text{inv}} = \left| \sum_{p \in \mathcal{P}_e(i)} \log P(M_{i,c}^p = 1) - \sum_{n \in \mathcal{N}_e(i)} \log P(M_{i,c}^n = 1) \right|,$$

which makes it equally probable that $\mathbf{z}_c^i$ is matched with its positive and negative pairs with respect to e. In other words, it makes $\mathbf{z}_c^i$ well-mixed with respect to e. This is analogous to adversarial approaches that train a discriminator to predict e, where the goal of representation learning is to fool the discriminator (Ganin et al., 2016; Edwards & Storkey, 2016). However, it can be difficult to apply these adversarial methods due to the complexity of minimax optimization. SCBD circumvents the need to train a discriminator, since the dot products between pairs of $\mathbf{z}_c$ can be used to predict e.

The fourth term in Equation 1 reconstructs $\mathbf{x}$ from $\mathbf{z}_c$ and $\mathbf{z}_s$. Importantly, we only use the reconstruction loss to optimize the decoder parameters, while holding $\mathbf{z}_c$ and $\mathbf{z}_s$ fixed. Therefore, the representation learning of $\mathbf{z}_c$ and $\mathbf{z}_s$ is done purely through supervised contrastive learning. It is possible to train the encoders and decoder jointly, which would likely improve the reconstruction quality. However, this adds the further complexity of balancing the relative contributions of the supervised contrastive and reconstruction losses by incorporating an additional hyperparameter. We leave this to future work, and focus on achieving strong performance on downstream tasks.

3 Improvements to variational approaches fall short of SCBD

As a basis of comparison for SCBD, we develop a block disentanglement approach based on Identifiable Variational Autoencoders (iVAEs) (Khemakhem et al., 2020). Several extensions to Variational Autoencoders (VAEs) (Kingma & Welling, 2014) address the problem of invariance to auxiliary variables, including the Variational Fair Autoencoder (Louizos et al., 2016) and the HSIC-constrained VAE (Lopez et al., 2018). These methods learn a single block of latent variables, and apply additional regularization to achieve invariance to an auxiliary variable. While successful in simple settings, these approaches have not gained widespread adoption for large-scale imaging data. Wang et al. (2023) applied contrastive learning to train a VAE with two blocks of latent variables, where one block does not condition on any auxiliary variables, and the other does. Our approach differs from theirs because we do not use contrastive learning, and both of our latent blocks condition on auxiliary variables.

We specify an iVAE with the same blocks of latent variables as SCBD. The generative model is defined as

$$p_\theta(\mathbf{x}, \mathbf{z}_c, \mathbf{z}_s \mid y, e) = p_\theta(\mathbf{x} \mid \mathbf{z}_c, \mathbf{z}_s) p_\theta(\mathbf{z}_c \mid y) p_\theta(\mathbf{z}_s \mid e),$$

while the inference model is defined as

$$q_\phi(\mathbf{z}_c, \mathbf{z}_s \mid \mathbf{x}, e) = q_\phi(\mathbf{z}_c \mid \mathbf{x}) q_\phi(\mathbf{z}_s \mid \mathbf{x}, e).$$

We fit this model by maximizing the evidence lower bound (ELBO) (Jordan et al., 1999), given by

$$\begin{aligned} &\min_{\theta,\phi} \mathbb{E}_{q_\phi(\mathbf{z}_c|\mathbf{x}) q_\phi(\mathbf{z}_s|\mathbf{x},e)}[-\log p_\theta(\mathbf{x} \mid \mathbf{z}_c, \mathbf{z}_s)] \\ &\quad + D_{KL}(q_\phi(\mathbf{z}_c \mid \mathbf{x}) \parallel p_\theta(\mathbf{z}_c \mid y)) + D_{KL}(q_\phi(\mathbf{z}_s \mid \mathbf{x}, e) \parallel p_\theta(\mathbf{z}_s \mid e)). \end{aligned}$$

Empirically, conditioning the posterior of $\mathbf{z}_s$ on both $\mathbf{x}$ and e, rather than just $\mathbf{x}$, significantly impacts downstream performance. We demonstrate this using ablation studies, comparing the two versions iVAE ($q_\phi(\mathbf{z}_s \mid \mathbf{x}, e)$) and iVAE ($q_\phi(\mathbf{z}_s \mid \mathbf{x})$). We hypothesize that conditioning the posterior of $\mathbf{z}_s$ on e makes it easier to encode the variation with respect to e in $\mathbf{z}_s$, which reduces the incentive to

encode it in $\mathbf{z}_c$. We use a mixture of experts approach to condition on e, where a neural network takes in $\mathbf{x}$ and outputs separate posterior parameters for each value of e.

Despite this improvement, we find that iVAE performs worse than SCBD in general. VAE-based block disentanglement methods inherently struggle to balance reconstruction and KL divergence minimization, leading to several failure modes. First, posterior collapse happens when the KL term is trivially minimized to zero by making the latent variables uninformative (Bowman et al., 2015; Razavi et al., 2019; Fu et al., 2019; Dai et al., 2020; Wang et al., 2021). Second, prior collapse occurs when learned parameters for $p_\theta(\mathbf{z}_c \mid y)$ collapse to the uninformative prior $p_\theta(\mathbf{z}_c)$. Third, numerical instability necessitates heuristics such as gradient clipping or skipping (Child, 2021). These issues significantly limit the ability to train VAEs with large-capacity neural networks, likely explaining why open-source VAE implementations rarely use generic image encoders like those in torchvision (Marcel & Rodriguez, 2010).

4 Experiments

We empirically validate SCBD on three datasets spanning two difficult real-world problems. We discuss domain generalization in Section 4.1, and batch correction in Section 4.2. In domain generalization, we use one synthetic and one realistic dataset, while in batch correction we use one large-scale realistic dataset with over 26 million images. The two problems are similar in that in both cases, we want $\mathbf{z}_c$ to represent the correlation between $\mathbf{x}$ and y that is invariant to e, and $\mathbf{z}_s$ to encode the remaining spurious correlations that depend on e. The key difference between the two problems relates to the evaluation. In domain generalization, we evaluate the ability to predict y given $\mathbf{z}_c$ on an out-of-distribution test set. In contrast, in batch correction the evaluation is in-distribution, and measures the degree to which $\mathbf{z}_c$ discards the information in e, while preserving the information in y.

4.1 Domain generalization

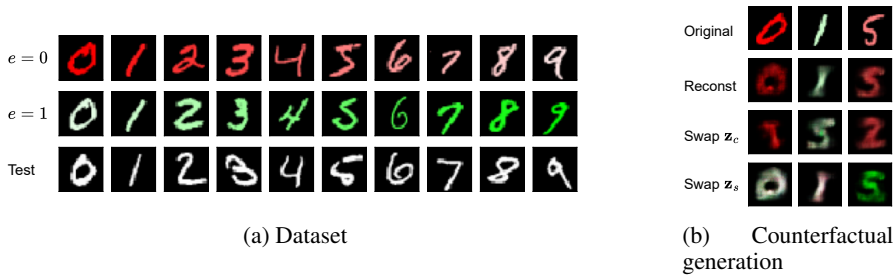
4.1.1 Problem description

Domain generalization is an out-of-distribution generalization problem, where the data come from different environments. Environments represent different conditions under which data are generated, such as the hospital that collected the samples. We assume data are sampled from a family of distributions $p_{\text{all}} = \{p_e(\mathbf{x}_e, y_e) : e \in \mathcal{E}_{\text{all}}\}$ indexed by the environment $e \in \mathcal{E}_{\text{all}} \subseteq \mathbb{N}$. The training data are sampled from $p_{\text{tr}} = \{p_e(\mathbf{x}_e, y_e) : e \in \mathcal{E}_{\text{tr}}\}$, where $\mathcal{E}_{\text{tr}} \subset \mathcal{E}_{\text{all}}$ is the set of training environments. The test data are sampled from $p_{\text{te}} = \{p_e(\mathbf{x}_e, y_e) : e \in \mathcal{E}_{\text{te}}\}$, where $\mathcal{E}_{\text{te}} \subset \mathcal{E}_{\text{all}}$ is the set of test environments. Because $\mathcal{E}_{\text{tr}}$ and $\mathcal{E}_{\text{te}}$ are disjoint, there is a distribution shift between p_{tr} and p_{te}. The goal is to predict y from $\mathbf{x}$ in a way that is invariant to e, so that we can generalize from p_{tr} to p_{te}.

4.1.2 In- and out-of-distribution performance must be negatively correlated

We begin by precisely characterizing the conditions under which SCBD should be effective at domain generalization. This is important, as it motivates our choice of datasets that we use in our experiments. The conditions are intuitively simple and empirically testable. SCBD requires the existence of spurious features in the training environments where the more they are learned, in-distribution performance improves, and out-of-distribution performance worsens. In other words, datasets need to exhibit a trade-off between in- and out-of-distribution performance. SCBD prevents the learning of such spurious features, since they are predictive of the training environments. This promotes the learning of features that are invariant to the environment, and thus generalize on the test environments.

We therefore want to evaluate SCBD on datasets that exhibit this trade-off. Fortunately, there is an empirical test for this, which is to train ERM across a large region of the hyperparameter search space, and check whether there are regions where in-distribution performance is strong, and is negatively correlated with out-of-distribution performance. Teney et al. (2024) carried out such a study, and found the trade-off to be particularly prominent on the Camelyon17-WILDS (Koh et al., 2021) dataset. We therefore include this dataset in our experiments.



(a) Dataset

(b) Counterfactual generation

Figure 2: Colored MNIST. (a) There is an environment-dependent correlation between color and digit on the training set, which does not persist on the test set where all digits are white. (b) We can generate images counterfactually using SCBD. When we swap $\mathbf{z}_c$ across examples, it changes the digit without affecting the color. In contrast, when we swap $\mathbf{z}_s$ across examples, it changes the color without affecting the digit. By composing digit and color independently, we generate images outside of the support of the training distribution, such as a light red one (bottom middle) and a bright green five (bottom right).

This trade-off between in- and out-of-distribution performance is the exception rather than the rule for domain generalization datasets. That is, despite the datasets being constructed to have qualitatively different training and test environments, it is often the case that in- and out-of-distribution performance are positively correlated. Wenzel et al. (2022) reached this conclusion by carrying out a large-scale empirical study involving 172 datasets, including those in the DomainBed (Gulrajani & Lopez-Paz, 2021) and WILDS (Koh et al., 2021) suites. It is difficult to outperform ERM when the correlation is positive, which may explain why Gulrajani & Lopez-Paz (2021) found ERM to be state-of-the-art across the DomainBed suite.

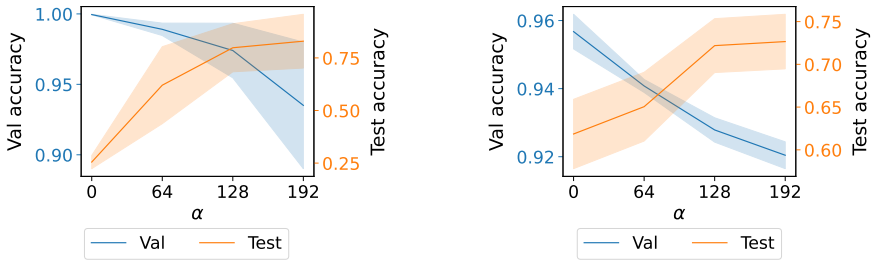
4.1.3 DATASETS

In addition to Camelyon17-WILDS, we experiment with one synthetic dataset. This dataset is called Colored MNIST (CMNIST), and extends the version from Arjovsky et al. (2019). The target label $y \in \{0, \dots, 9\}$ represents the digit. There are two training environments and a test environment. In the training environments $e \in \{0, 1\}$, there is an environment-dependent correlation between the color and y (Figure 2a). For $e = 0$ the color changes from dark to light red as the digit increases. In contrast, for $e = 1$ the color changes from light to dark green as the digit increases. All digits are white in the test environment. This presents a severe distribution shift, since color is perfectly predictive of y in the training environments, but is unpredictive in the test environment. Details regarding the data generating process are in Appendix A.2.1. We train ERM across a range of learning rates and maximum training steps on this dataset, and observe that in- and out-of-distribution performance are negatively correlated (Appendix Figure 5). This satisfies the assumptions of SCBD, so therefore we expect it to be effective on this dataset. We expect $\mathbf{z}_c$ to encode the digit, and $\mathbf{z}_s$ to encode the environment-specific colors.

Camelyon17-WILDS (Koh et al., 2021) is a patch-based variant of the original Camelyon17 dataset (Bándi et al., 2019) of histopathology images of breast tissue, and represents a binary classification task of predicting the presence of a tumor. The data were collected in five hospitals, and have significant inter-hospital batch effects. It has been reported that for similar datasets, the most significant batch effects are from differences in how the slides are stained (Tellez et al., 2019). As mentioned previously, Teney et al. (2024) showed that this dataset exhibits a trade-off between in- and out-of-distribution performance, and therefore satisfies the assumptions of SCBD. We also verify this in Appendix Figure 11. On this dataset, we want $\mathbf{z}_c$ to represent the biomarkers of disease that are invariant across hospitals, and $\mathbf{z}_s$ to represent the hospital-specific spurious correlations.

4.1.4 BASELINES

We compare SCBD to a diverse range of algorithms that are considered to be standard baselines in the domain generalization literature. This includes Empirical Risk Minimization (ERM) (Vapnik, 1995), CORrelation ALignment (CORAL) (Sun & Saenko, 2016), Domain-Adversarial Neural



(a) CMNIST (b) Camelyon17-WILDS

Figure 3: Increasing α strengthens the degree that $\mathbf{z}_c$ is invariant to e, and monotonically improves test accuracy at the expense of validation accuracy.

Table 1: Test accuracy (%) for domain generalization for ten random seeds. SCBD with $\alpha = 192$ significantly outperforms all baselines on both CMNIST and Camelyon17-WILDS.

Algorithm	CMNIST	Camelyon17-WILDS
SCBD ($\alpha = 0$)	25.5 ± 3.0	61.9 ± 3.8
SCBD ($\alpha = 192$)	$\mathbf{82.9 \pm 12.1}$	$\mathbf{72.7 \pm 3.0}$
ERM	37.8 ± 2.6	65.8 ± 4.9
CORAL	37.6 ± 3.6	59.5 ± 7.7
DANN	39.0 ± 4.5	55.2 ± 6.7
IRM	37.0 ± 4.2	66.3 ± 2.1
Fish	48.2 ± 3.5	49.1 ± 0.9
Group DRO	35.0 ± 2.9	68.4 ± 7.3
iVAE ($q_\phi(\mathbf{z}_s \mid \mathbf{x}, e)$)	52.1 ± 37.6	52.0 ± 2.0
iVAE ($q_\phi(\mathbf{z}_s \mid \mathbf{x})$)	37.7 ± 29.4	51.9 ± 4.3

Networks (DANN) (Ganin et al., 2016), Invariant Risk Minimization (IRM) (Arjovsky et al., 2019), Fish (Shi et al., 2022), and Group Distributionally Robust Optimization (Group DRO) (Sagawa et al., 2020). We additionally compare against our two versions of iVAE from Section 3 for completeness.

4.1.5 Qualitative results

Our image generation results in Figure 2b qualitatively demonstrate that SCBD achieves block disentanglement. This is possible on CMNIST because we know that the ground-truth phenomenon of interest is the digit, and the spurious correlation is the color. These results show that when we swap $\mathbf{z}_c$ between examples, it changes the digit without affecting the color. In contrast, when we swap $\mathbf{z}_s$ between examples, it changes the color without affecting the digit. Note that the quality of the reconstructed images is relatively poor because, as mentioned in Section 2, the decoder is not trained jointly with the encoders. We leave it to future work to train the decoder jointly and improve the image reconstruction capability of SCBD. We provide similar visualization results with the iVAE in Appendix Figure 6.

4.1.6 Quantitative results

We present two kinds of quantitative results. In Figure 3, we show that by increasing α, SCBD removes spurious correlations that are specific to the training environments. This encourages the learning of features that are invariant to the environment, which yields a smooth and monotonic trade-off between in- and out-of-distribution performance on both datasets.

In Table 1, we show the test accuracy on both datasets for SCBD and the baseline algorithms. We report the average and standard deviation for ten random seeds. For the baseline algorithms, we optimize the hyperparameters with respect to the performance on the in-distribution validation set. The hyperparameter search space for each algorithm is provided in Appendix Table 5. Most of the baseline results for Camelyon17-WILDS are taken from the authors' leaderboard, with the

exception of Fish (Shi et al., 2022), which we evaluate ourselves. Our results for Fish are weaker than those reported on the leaderboard, because we additionally included the pretraining duration in the hyperparameter search space. The leaderboard results used the value of this hyperparameter that achieved the best test accuracy, as described in the appendix of Shi et al. (2022).

For SCBD, we apply the same model selection procedure to optimize the learning rate and weight decay. We do not optimize α during model selection, since this would result in choosing $\alpha = 0$. We report the test accuracy for $\alpha = 0$ and $\alpha = 192$ as evidence that the invariance loss in SCBD is effective at removing spurious correlations and improving out-of-distribution performance. With $\alpha = 192$, SCBD significantly outperforms all baseline algorithms across both datasets. Tuning α corresponds to model selection with respect to an unknown test distribution, which is a difficult open problem (Gulrajani & Lopez-Paz, 2021), and is a limitation shared by other works (Makino et al., 2022a; Wortsman et al., 2022).

Also, we demonstrate the robustness of our approach to the choice of hyperparameters by providing the results of ablation studies in Appendix A.2.1 and A.2.2, where we vary $D_{\mathbf{z}_c}$ and $D_{\mathbf{z}_s}$, the batch size, and the degree of weight decay.

We additionally experiment with PACS (Li et al., 2017) and VLCS (Fang et al., 2013) from DomainBed (Gulrajani & Lopez-Paz, 2021), and include the results in Appendix Sections A.2.3 and A.2.4. We find that these datasets exhibit a positive correlation between in- and out-of-distribution performance, which is consistent with Wenzel et al. (2022). Since this violates the assumptions of SCBD, we are unable to trade-off in- and out-of-distribution performance, as expected.

4.2 Batch correction with a real-world Optical Pooled Screen dataset

4.2.1 Problem description and dataset

Having demonstrated the efficacy of SCBD on domain generalization, we proceed to a related but different problem called batch correction. Here, we experiment with one realistic single-cell perturbation dataset that is of significantly large scale. We use the Optical Pooled Screen (OPS) (Feldman et al., 2019) dataset from Funk et al. (2022) comprised of 26 million images of single cells, each perturbed with one of $5{,}050$ genetic perturbations targeting an expressed gene, including one non-targeting control. Such data are collected in order to understand the effect of each perturbation on cellular morphology. The 100×100 pixel images have four channels that measure staining information for key cellular features: DNA damage, F-actin, DNA content, and microtubules. Each channel therefore measures a unique aspect of a cell's phenotype, which taken together shed light on how each perturbed genes affects the cell. An important problem in the field is to build a cartography of perturbation effects on cells, by grouping perturbed genes by their similarity on the phenotypic level (Celik et al., 2024). This perturbation map is then interpreted to characterize the function of unknown genes, recapitulate protein complexes, and highlight interacting pathways (Rood et al., 2024).

OPS technologies generate large quantities of data in a cost effective manner by conducting several batches of experiments in parallel. In this case, the data were collected at a single lab but using 34 wells. There can be significant unintended variation across wells, solely based on seemingly minor differences in experimental conditions. For example, if the wells are stained sequentially, the difference in elapsed time can result in different image brightness across wells. Our goal with SCBD is to capture this unintended variation across wells in $\mathbf{z}_s$, so that $\mathbf{z}_c$ is an unconfounded representation of the impact of genetic perturbations on cell morphology.

For each image of a single cell $\mathbf{x}$, y labels the genetic perturbation that was applied, and e labels which of the 34 wells the cell was in. By optimizing the SCBD objective in Equation 1, we ensure that the variation in the images due to the perturbation is represented by $\mathbf{z}_c$, and the variation due to the well e is represented by $\mathbf{z}_s$. We can then use $\mathbf{z}_c$ for downstream analysis. For this task, we are using $\mathbf{z}_c$ with $D_{\mathbf{z}_c} = D_{\mathbf{z}_s} = 64$, whereas we used $\mathbf{r}_c$ for domain generalization. This is because all of our baselines for this task use 64 dimensional embeddings, so the lower-dimensional $\mathbf{z}_c$ helps ensure a fair comparison. We show results using ResNet-18 encoders in the main text, and additionally provide results using DenseNet-121 encoders in Appendix Figure 25.

We evaluate two tasks to understand the degree to which we remove the influence of e, while preserving the information in y. We describe the tasks at a high level here, and provide the details in

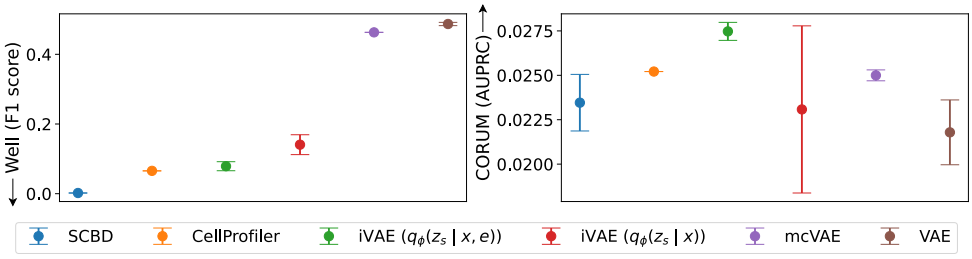


Figure 4: Comparison of SCBD to CellProfiler and VAE-based baselines on real-world batch correction. Left: Performance of predicting the well label e. Right: Performance on predicting protein complex membership (biological content). SCBD is almost completely unpredictive of the well, while retaining a similar level of biological information as CellProfiler.

Appendix A.3. The first task is to use the perturbation embeddings to predict e, which measures the sensitivity to inter-well batch effects. We fit a linear classifier on top of each of the embeddings, and compute the F1 score. We want the performance on this task to be weak.

The second task is CORUM prediction, which is one measure of the biological information content in the embeddings. This task relies on the CORUM database (Ruepp et al., 2010) as the ground truth of whether two genes are functionally related based on their membership in the same protein complex (a definition previously used in the context of this biological screen). We take the biological embeddings corresponding to those genetic perturbations y, interpret their dot product as the prediction that they are similar, and use these predictions to compute the area under the precision-recall curve. Unlike the first task, we want the performance on this task to be strong.

4.2.2 Baselines

CellProfiler (Carpenter et al., 2006) is the most important baseline that we compare SCBD against. It is an open-source software that takes in an image of a cell, and outputs several thousand manually-engineered morphological features that describe the cell's phenotype. It is a very strong baseline in which substantial human-expert effort has been invested, and its representations are post-processed to effectively remove the variation across plates and wells. Following conventional practice, we use the top-64 principal components of the full set of CellProfiler features.

The remaining baselines are all based on VAEs, which are considered conventional. We experiment with iVAE ($q_\phi(\mathbf{z}_s \mid \mathbf{x}, e)$) and iVAE ($q_\phi(\mathbf{z}_s \mid \mathbf{x})$) from Section 3, as well as the Multi-Contrastive VAE (mcVAE) (Wang et al., 2023), which uses two blocks of latent variables in order to represent the perturbation effect and the natural cell-to-cell variation. While it was previously shown that mcVAE is effective at modeling genetic perturbations, it has a significant weakness in that it does not effectively correct for batch effects. Finally, for our simplest baseline we use a vanilla VAE (Kingma & Welling, 2014), which has a single block of latent variables, and ignores y and e. For all VAE-based models, we use 64 dimensional latent variables in each block. The perturbation embedding is $\mathbf{z}_c$ for SCBD and the iVAEs. For mcVAE it is the block of salient variables, and for CellProfiler and the vanilla VAE, there is only a single block of latent variables.

4.2.3 Quantitative results

We show our results on both tasks in Figure 4. SCBD achieves close to zero predictive performance on the well-prediction task, while retaining a similar level of biological information as CellProfiler. Thus, $\mathbf{z}_c$ estimated with SCBD can be used by biologists for downstream analysis, and they can be confident that any conclusions reached are not due to the inter-well variation. Our results also show that for the iVAE baselines, additionally conditioning the posterior of $\mathbf{z}_s$ on e significantly improves the results on both tasks. Although mcVAE performs better than the vanilla VAE on CORUM due to its ability to incorporate the perturbation labels y, they are both highly susceptible to the inter-well batch effects. This highlights the fact that explicit regularization is required in order to purge the effect of e from the embeddings, and that this does not occur naturally.

5 Related work

5.1 Disentangled representation learning

Our goal of block disentanglement is closely related to that of disentangled representation learning, which assumes that a relatively small number of independent factors are sufficient to explain the important patterns of variation in $\mathbf{x}$. Disentangled representation learning is typically cast as learning a latent variable $\mathbf{z} \in \mathbb{R}^{D_\mathbf{z}}$, where $\mathbf{z}$ is disentangled if its individual components $z_1, \ldots, z_{D_\mathbf{z}}$ are independent and semantically meaningful (Higgins et al., 2017; Esmaeili et al., 2019; Kim & Mnih, 2018; Chen et al., 2018). This informal definition of disentanglement is generally agreed upon, and it is not trivial to define this concept quantitatively (Eastwood & Williams, 2018; Higgins et al., 2018). This is related to independent component analysis (Comon, 1994; Jutten & Herault, 1991; Hyvärinen & Oja, 2000), which makes the additional assumption that the encoding is noiseless.

With block disentanglement, instead of assuming there are $D_\mathbf{z}$ independent scalar factors, we assume there are two independent vector-valued factors $\mathbf{z}_c \in \mathbb{R}^{D_{\mathbf{z}_c}}$ and $\mathbf{z}_s \in \mathbb{R}^{D_{\mathbf{z}_s}}$. Recent works study identifiability for block disentanglement (Von Kügelgen et al., 2021; Lachapelle & Lacoste-Julien, 2022; Kong et al., 2022; Lachapelle et al., 2024; Lopez et al., 2024). While we believe this is an important research direction, we focus on developing a simple algorithm that achieves strong empirical results on difficult real-world problems.

5.2 Invariant representation learning

The challenge of domain generalization has gained significant attention as ML systems often fail to generalize out-of-distribution. Peters et al. (2016) introduced a framework for causal inference using invariant prediction, helping maintain predictive accuracy under interventions or environmental changes. Building on this foundation, Arjovsky et al. (2019) proposed IRM, a learning paradigm for learning an embedding of the data representation such that the optimal classifier on top of that representation remains invariant across different environments. These works, as well as many extensions (Lu et al., 2021), have been benchmarked on datasets created by the research community, such as those in the DomainBed (Gulrajani & Lopez-Paz, 2021) and WILDS (Koh et al., 2021) suites. Gulrajani & Lopez-Paz (2021) revealed that with rigorous model selection, ERM often achieves state-of-the-art performance, challenging the perceived benefits of more complex domain generalization methods.

6 Conclusion

We presented Supervised Contrastive Block Disentanglement (SCBD), an algorithm for block disentanglement that is based purely on supervised contrastive learning. We use SCBD to estimate $\mathbf{z}_c$ such that it represents the correlation between $\mathbf{x}$ and y that is invariant to e. This invariance, which is considered difficult to achieve in practice, allows us to solve two difficult real-world problems. The first is domain generalization, where we achieve strong out-of-distribution generalization on a synthetic dataset called Colored MNIST, as well as a real-world histopathology dataset called Camelyon17-WILDS. The second is batch correction, where we use SCBD to learn representations of single-cell perturbations from over 26 million images that are nearly free of inter-well batch effects.

We believe a promising direction for future work is to investigate how to jointly train the decoder to combine the representation learning capabilities of supervised contrastive learning and generative modeling. While iVAE failed at domain generalization, it achieved strong performance on the CORUM task in the batch correction problem. We interpret this as a sign that image reconstruction can yield additional useful features that SCBD is currently not capturing. With improved generative modeling, SCBD has the potential to be used for impactful counterfactual image generation, such as generating images of the same cell under different perturbations. Also, in this work we assumed access to the variable e, which labels the source of unwanted variation. We leave it to future work to learn this variable from data.

REFERENCES

John Arevalo, Ellen Su, Jessica D Ewald, Robert van Dijk, Anne E Carpenter, and Shantanu Singh. Evaluating batch correction methods for image-based cell profiling. *Nature Communications*, 2024.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv*, 2019.

Péter Bándi, Oscar Geessink, Quirine Manson, Marcory Van Dijk, Maschenka Balkenhol, Meyke Hermsen, Babak Ehteshami Bejnordi, Byungjae Lee, Kyunghyun Paeng, et al. From detection of individual metastases to classification of lymph node status at the patient level: The CAME-LYON17 challenge. *IEEE Trans. Medical Imaging*, 2019.

Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. *Advances in Neural Information Processing Systems*, 2011.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv*, 2015.

Anne E Carpenter, Thouis R Jones, Michael R Lamprecht, Colin Clarke, In Han Kang, Ola Friman, David A Guertin, Joo Han Chang, Robert A Lindquist, Jason Moffat, et al. Cellprofiler: image analysis software for identifying and quantifying cell phenotypes. *Genome Biology*, 2006.

Safiye Celik, Jan-Christian Hütter, Sandra Melo Carlos, Nathan H Lazar, Rahul Mohan, Conor Tillinghast, Tommaso Biancalani, Marta M Fay, Berton A Earnshaw, and Imran S Haque. Building, benchmarking, and exploring perturbative maps of transcriptional and morphological data. *bioRxiv*, 2024.

Srinivas Niranj Chandrasekaran, Beth A Cimini, Amy Goodale, Lisa Miller, Maria Kost-Alimova, Nasim Jamali, John G Doench, Briana Fritchman, Adam Skepner, Michelle Melanson, et al. Three million images and morphological profiles of cells treated with matched chemical and genetic perturbations. *Nature Methods*, 2024.

Ricky TQ Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. *Advances in Neural Information Processing Systems*, 2018.

Rewon Child. Very deep VAEs generalize autoregressive models and can outperform them on images. In *International Conference on Learning Representations*, 2021.

Pierre Comon. Independent component analysis, a new concept? *Signal Processing*, 1994.

Bin Dai, Ziyu Wang, and David Wipf. The usual suspects? reassessing blame for VAE posterior collapse. In *International Conference on Machine Learning*, 2020.

Cian Eastwood and Christopher KI Williams. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, 2018.

Harrison Edwards and Amos Storkey. Censoring representations with an adversary. In *International Conference on Learning Representations*, 2016.

Babak Esmaeili, Hao Wu, Sarthak Jain, Alican Bozkurt, Narayanaswamy Siddharth, Brooks Paige, Dana H Brooks, Jennifer Dy, and Jan-Willem Meent. Structured disentangled representations. In *International Conference on Artificial Intelligence and Statistics*, 2019.

Chen Fang, Ye Xu, and Daniel N Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *International Conference on Computer Vision*, 2013.

David Feldman, Avtar Singh, Jonathan L Schmid-Burgk, Rebecca J Carlson, Anja Mezger, Anthony J Garrity, Feng Zhang, and Paul C Blainey. Optical pooled screens in human cells. *Cell*, 2019.

Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. Cyclical annealing schedule: A simple approach to mitigating KL vanishing. *arXiv*, 2019.

Luke Funk, Kuan-Chung Su, Jimmy Ly, David Feldman, Avtar Singh, Brittania Moodie, Paul C Blainey, and Iain M Cheeseman. The phenotypic landscape of essential human genes. *Cell*, 2022.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 2016.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *International Conference on Learning Representations*, 2021.

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*, 2016.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv*, 2016.

Irina Higgins, Loic Matthey, Arka Pal, Christopher P Burgess, Xavier Glorot, Matthew M Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.

Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, and Alexander Lerchner. Towards a definition of disentangled representations. *arXiv*, 2018.

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*, 2017.

Aapo Hyvärinen and Erkki Oja. Independent component analysis: algorithms and applications. *Neural Networks*, 2000.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 1999.

Christian Jutten and Jeanny Herault. Blind separation of sources, part i: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, 1991.

Ilyes Khemakhem, Diederik Kingma, Ricardo Monti, and Aapo Hyvarinen. Variational autoencoders and nonlinear ICA: A unifying framework. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, 2020.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *International Conference on Machine Learning*, 2018.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

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*, 2021.

Lingjing Kong, Shaoan Xie, Weiran Yao, Yujia Zheng, Guangyi Chen, Petar Stojanov, Victor Akinwande, and Kun Zhang. Partial disentanglement for domain adaptation. In *International Conference on Machine Learning*, 2022.

Sébastien Lachapelle and Simon Lacoste-Julien. Partial disentanglement via mechanism sparsity. *arXiv*, 2022.

Sébastien Lachapelle, Divyat Mahajan, Ioannis Mitliagkas, and Simon Lacoste-Julien. Additive decoders for latent variables identification and cartesian-product extrapolation. *Advances in Neural Information Processing Systems*, 2024.

Jeffrey T Leek, Robert B Scharpf, Héctor Corrada Bravo, David Simcha, Benjamin Langmead, W Evan Johnson, Donald Geman, Keith Baggerly, and Rafael A Irizarry. Tackling the widespread and critical impact of batch effects in high-throughput data. *Nature Reviews Genetics*, 2010.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *International Conference on Computer Vision*, 2017.

Romain Lopez, Jeffrey Regier, Michael I. Jordan, and Nir Yosef. Information constraints on auto-encoding variational bayes. In *Advances in Neural Information Processing Systems*, 2018.

Romain Lopez, Jan-Christian Huetter, Ehsan Hajiramezanali, Jonathan K Pritchard, and Aviv Regev. Toward the identifiability of comparative deep generative models. In *Conference on Causal Learning and Reasoning*, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Christos Louizos, Kevin Swersky, Yujia Li, Max Welling, and Richard S. Zemel. The variational fair autoencoder. In *International Conference on Learning Representations*, 2016.

Chaochao Lu, Yuhuai Wu, José Miguel Hernández-Lobato, and Bernhard Schölkopf. Invariant causal representation learning for out-of-distribution generalization. In *International Conference on Learning Representations*, 2021.

Taro Makino, Krzysztof Geras, and Kyunghyun Cho. Generative multitask learning mitigates target-causing confounding. *Advances in Neural Information Processing Systems*, 2022a.

Taro Makino, Stanislaw Jastrzebski, Witold Oleszkiewicz, Celin Chacko, Robin Ehrenpreis, Naziya Samreen, Chloe Chhor, Eric Kim, Jiyon Lee, Kristine Pysarenko, Beatriu Reig, Hildegard Toth, Divya Awal, Linda Du, Alice Kim, James Park, Daniel K. Sodickson, Laura Heacock, Linda Moy, Kyunghyun Cho, and Krzysztof J. Geras. Differences between human and machine perception in medical diagnosis. *Scientific Reports*, 2022b.

Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In *International Conference on Multimedia*, 2010.

Scott Mayer McKinney, Marcin Sieniek, Varun Godbole, Jonathan Godwin, Natasha Antropova, Hutan Ashrafian, Trevor Back, Mary Chesus, Greg S Corrado, Ara Darzi, et al. International evaluation of an AI system for breast cancer screening. *Nature*, 2020.

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, 2013.

Tan H Nguyen, Dinkar Juyal, Jin Li, Aaditya Prakash, Shima Nofallah, Chintan Shah, Sai Chowdary Gullapally, Limin Yu, Michael Griffin, Anand Sampat, et al. Contrimix: Scalable stain color augmentation for domain generalization without domain labels in digital pathology. *arXiv*, 2023.

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)*, 2016.

Ali Razavi, Aäron van den Oord, Ben Poole, and Oriol Vinyals. Preventing posterior collapse with delta-vaes. *arXiv*, 2019.

Jennifer E Rood, Anna Hupalowska, and Aviv Regev. Toward a foundation model of causal cell and tissue biology with a perturbation cell and tissue atlas. *Cell*, 2024.

Andreas Ruepp, Brigitte Waegele, Martin Lechner, Barbara Brauner, Irmtraud Dunger-Kaltenbach, Gisela Fobo, Goar Frishman, Corinna Montrone, and H-Werner Mewes. CORUM: the comprehensive resource of mammalian protein complexes—2009. *Nucleic Acids Research*, 2010.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 1986.

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

Yuge Shi, Jeffrey Seely, Philip HS Torr, N Siddharth, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. Gradient matching for domain generalization. In *International Conference on Learning Representations*, 2022.

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

David Tellez, Geert Litjens, Péter Bándi, Wouter Bulten, John-Melle Bokhorst, Francesco Ciompi, and Jeroen Van Der Laak. Quantifying the effects of data augmentation and stain color normalization in convolutional neural networks for computational pathology. *Medical Image Analysis*, 2019.

Damien Teney, Yong Lin, Seong Joon Oh, and Ehsan Abbasnejad. ID and OOD performance are sometimes inversely correlated on real-world datasets. *Advances in Neural Information Processing Systems*, 2024.

Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

Julius Von Kügelgen, Yash Sharma, Luigi Gresele, Wieland Brendel, Bernhard Schölkopf, Michel Besserve, and Francesco Locatello. Self-supervised learning with data augmentations provably isolates content from style. *Advances in Neural Information Processing Systems*, 2021.

Yixin Wang, David Blei, and John P Cunningham. Posterior collapse and latent variable non-identifiability. *Advances in Neural Information Processing Systems*, 2021.

Zitong Jerry Wang, Romain Lopez, Jan-Christian Hütter, Takamasa Kudo, Heming Yao, Philipp Hanslovsky, Burkhard Hoeckendorf, Rahul Moran, David Richmond, and Aviv Regev. Multi-contrastivevae disentangles perturbation effects in single cell images from optical pooled screens. *bioRxiv*, 2023.

Florian Wenzel, Andrea Dittadi, Peter Gehler, Carl-Johann Simon-Gabriel, Max Horn, Dominik Zietlow, David Kernert, Chris Russell, Thomas Brox, Bernt Schiele, et al. Assaying out-of-distribution generalization in transfer learning. *Advances in Neural Information Processing Systems*, 2022.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. Robust fine-tuning of zero-shot models. In *Conference on Computer Vision and Pattern Recognition*, 2022.

APPENDIX

A EXPERIMENTS

A.1 EXPERIMENTAL SETUP

All of our experiments were done using a single NVIDIA A100 GPU on our institutions' high-performance computing clusters.

A.1.1 SUPERVISED CONTRASTIVE BLOCK DISENTANGLEMENT

Our experimental setup for SCBD is remarkably similar across all of our experiments, which highlights the generality of our approach. We resize the images to 32×32 pixels and use a batch size of $2{,}048$. We set the temperature parameter in the supervised contrastive losses to $\tau = 0.1$. We adopt both of these practices from Khosla et al. (2020). For optimization, we use AdamW (Loshchilov & Hutter, 2019) with 1×10^{-4} learning rate and 0.01 weight decay. We chose these values because they resulted in stable training and validation curves across our experiments, and did not tune them extensively.

For domain generalization, we set $D_{\mathbf{z}_c} = D_{\mathbf{z}_s} = 128$, which we adopt from Khosla et al. (2020). We train for a maximum of $25{,}000$ steps, and select the weights that minimize the validation loss. We obtain error bars by repeating each experiment with ten random seeds.

For batch correction, we set $D_{\mathbf{z}_c} = D_{\mathbf{z}_s} = 64$ in order to ensure a fair comparison with the top-64 PCA features of CellProfiler. To sample a minibatch, we first sample 256 distinct values of y from the class distribution of the training set, and then sample the same number of examples per value of y. This was necessary in order to ensure a large number of positive pairs with respect to y in our supervised contrastive losses, given that there are $5{,}050$ classes. We trained for a maximum of $150{,}000$ steps, and evaluated on the test set using the weights that minimize the validation loss. We obtain error bars by repeating each experiment with three random seeds.

We use standard architectures such as ResNet-18 (He et al., 2016) and DenseNet-121 (Huang et al., 2017) for the encoders $\mathrm{Enc}_c(\mathbf{x})$ and $\mathrm{Enc}_s(\mathbf{x})$. The projection networks $\mathrm{Proj}_c(\mathbf{r}_c)$ and $\mathrm{Proj}_s(\mathbf{r}_s)$ are two-layer Multilayered Perceptrons (Rumelhart et al., 1986) with hidden sizes of $D_{\mathbf{r}_c}$ and $D_{\mathbf{r}_s}$, and GELU activations (Hendrycks & Gimpel, 2016). Our decoder $\mathrm{Dec}(\mathbf{z}_c, \mathbf{z}_s)$ architecture is shown in Appendix Table 2, with GELU activations (Hendrycks & Gimpel, 2016) between layers. We use an additive decoder (Lachapelle et al., 2024), and found this to be necessary to achieve sensible visualization results on CMNIST. That is, we define

$$\log p(\mathbf{x} \mid \mathbf{z}_c, \mathbf{z}_c) = \mathrm{Dec}_c(\mathbf{z}_c) + \mathrm{Dec}_s(\mathbf{z}_s),$$

where both Dec_c and Dec_s have the same architecture.

Table 2: SCBD decoder architecture

```
Linear(64, 256 * (2 ** 2))
ConvTranspose2d(256, 256, 3, stride=2, padding=1, output_padding=1)
Conv2d(256, 256, 3, padding=1)
ConvTranspose2d(256, 256, 3, stride=2, padding=1, output_padding=1)
Conv2d(256, 256, 3, padding=1)
ConvTranspose2d(256, 256, 3, stride=2, padding=1, output_padding=1)
Conv2d(256, 256, 3, padding=1)
ConvTranspose2d(256, 256, 3, stride=2, padding=1, output_padding=1)
Conv2d(256, 128, 3, padding=1)
Conv2d(128, img_ch, 1)
```

A.1.2 VARIATIONAL AUTOENCODERS

For our experiments with VAE-based approaches, we use the same experimental setup used in Wang et al. (2023), including the architecture and hyperparameters. We resize the images to 64×64 pixels

and use a batch size of 1024. The encoder and decoder architectures are in Appendix Tables 3 and 4, with GELU activations (Hendrycks & Gimpel, 2016) between layers. Since the CMNIST images are 32×32, we modify the architectures to reduce the up- and down-sampling. For optimization, we use the AdamW (Loshchilov & Hutter, 2019) optimizer with 1×10^{-4} learning rate and 0.01 weight decay. We additionally skip gradients with a norm above 1×10^{12}, and clip gradients with a norm above 1×10^{6}, as done in Child (2021). We train for a maximum of $50{,}000$ steps for domain generalization, and three epochs for batch correction, and select the weights with minimum validation loss. We report the validation and test performance across ten random seeds for domain generalization, and three random seeds for batch correction.

Table 3: VAE encoder architecture

```
Conv2d(img_c, 32, 3, stride=2, padding=1)
Conv2d(32, 32, 3, padding=1)
Conv2d(32, 64, 3, stride=2, padding=1)
Conv2d(64, 64, 3, padding=1)
Conv2d(64, 64, 3, stride=2, padding=1)
Linear(64 * (8 ** 2), 2 * 64)
```

Table 4: VAE decoder architecture

```
Linear(2 * 64, 64 * (8 ** 2))
ConvTranspose2d(64, 64, 3, stride=2, padding=1, out_padding=1)
Conv2d(64, 64, 3, padding=1)
ConvTranspose2d(64, 32, 3, stride=2, padding=1, out_padding=1)
Conv2d(32, 32, 3, padding=1)
ConvTranspose2d(32, img_c, 3, stride=2, padding=1, out_padding=1)
```

A.1.3 OTHER BASELINES

Table 5: Hyperparameter search space

Condition	Hyperparameter	Search space
CMNIST	Learning rate	{0.0001, 0.001, 0.01}
	Weight decay	{0, 0.001, 0.01}
	Batch size	{32}
	Maximum epochs	{1, 20, 100}
Camelyon17-WILDS	Learning rate	{0.0001, 0.001, 0.01}
	Weight decay	{0, 0.001, 0.01}
	Batch size	{32}
	Maximum epochs	{5}
CORAL	Penalty weight	{0.1, 1, 10}
DANN	Penalty weight	{0.1, 1, 10}
IRM	Penalty weight	{1, 10, 100, 1000}
Fish	Pretrain steps	{1000, 10000}
	Meta learning rate	{0.001, 0.01, 0.1}
Group DRO	Step size	{0.01}

A.2 DOMAIN GENERALIZATION

A.2.1 COLORED MNIST

Data generating process The images are 32×32 pixels, and are RGB. There are two training environments and a test environment. In the first training environment, which we label $e = 0$, we set the foreground pixels in the red channel to the value one, and those in the green and blue channels to the value $y/|\mathcal{Y}|$, where $|\mathcal{Y}| = 10$ is the number of digits. For images with the digit zero, $y = 0$, so the digit is colored completely red. Then, as the digit increases from zero to nine, the digits are colored red, but with a decreasing intensity. In the second training environment, which we label $e = 1$, we set the foreground pixels in the green channel to the value one, and those in the red and blue channels to the value $(|\mathcal{Y}| - 1 - y)/|\mathcal{Y}|$. This has the effect of the digits being colored green, where the intensity increases with as the digit increases from zero to nine. In the test environment, the foreground pixels are set to one in all channels, which makes all of the digits white.

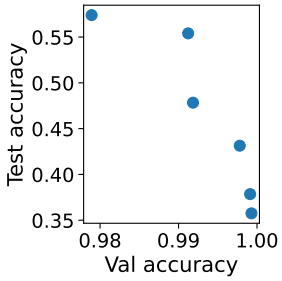


Figure 5: In- and out-of-distribution performance are negatively correlated on CMNIST, which satisfies the assumptions made by SCBD.

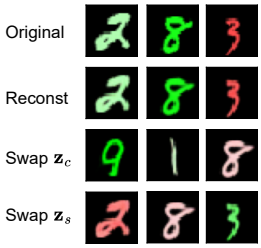


Figure 6: Counterfactual generation with iVAE. When we swap $\mathbf{z}_c$, it changes the digit but not the color, and when we swap $\mathbf{z}_s$, it changes the color but not the digit. iVAE generates better-looking images than SCBD, since the decoder is trained jointly with the encoder for iVAE.

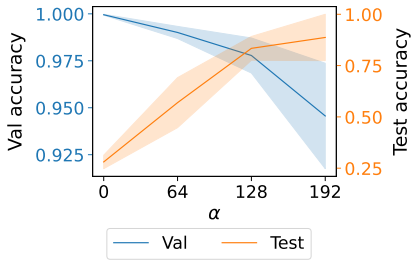


Figure 7: CMNIST results for an ablation in which we omit $\mathbf{z}_s$, and learn a single block of latent variables $\mathbf{z}_c$ that are correlated with y and invariant to e. These results are similar to the model that learns $\mathbf{z}_s$. We use ResNet-18 encoders here, as we did in the main text.

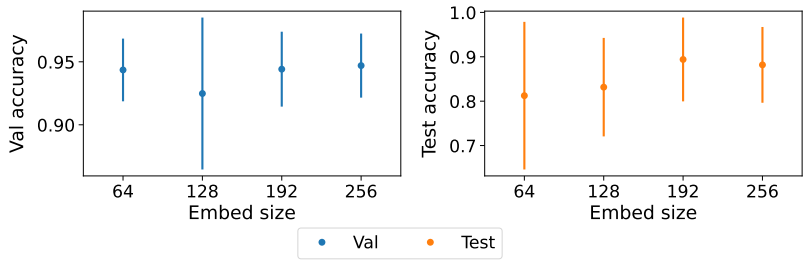


Figure 8: CMNIST embedding size ($D_{\mathbf{z}_c}$ and $D_{\mathbf{z}_s}$) ablation study for SCBD with ResNet-18 encoders and $\alpha = 192$. The results are relatively consistent across different embedding sizes.

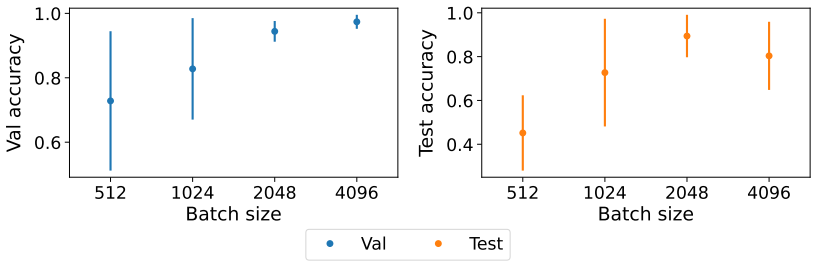


Figure 9: CMNIST batch size ablation study for SCBD with ResNet-18 encoders and $\alpha = 192$. The results are generally better for larger batch sizes, which was also observed by the authors of SCL (Khosla et al., 2020).

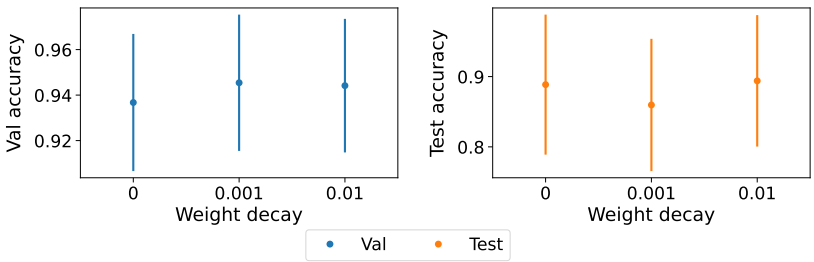


Figure 10: CMNIST weight decay ablation study for SCBD with ResNet-18 encoders and $\alpha = 192$. The results are relatively consistent across different degrees of weight decay.

A.2.2 CAMELYON17-WILDS

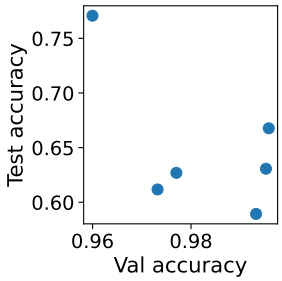


Figure 11: In- and out-of-distribution performance are negatively correlated on Camelyon17-WILDS. This is consistent with Teney et al. (2024), and therefore this dataset satisfies the assumptions made by SCBD.

.

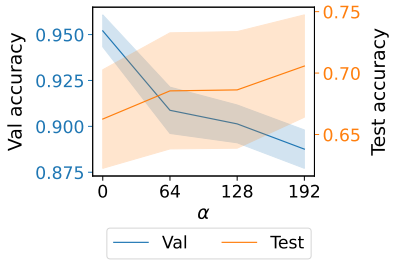
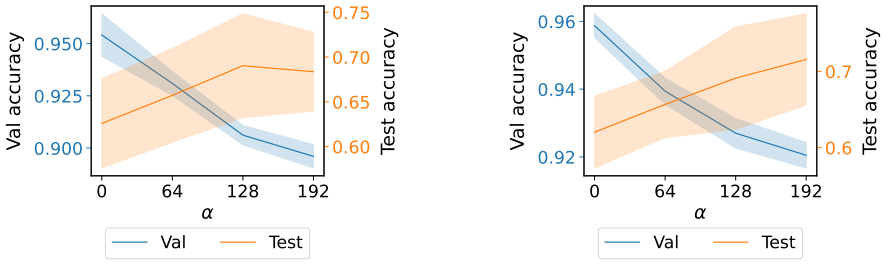


Figure 12: Camelyon17-WILDS results for SCBD using ResNet-18 encoders. The conclusions are the same as with the DenseNet-121 encoders.



(a) ResNet-18 (b) DenseNet-121

Figure 13: Camelyon17-WILDS results for an ablation in which we omit $\mathbf{z}_s$, and learn a single block of latent variables $\mathbf{z}_c$ that are correlated with y and invariant to e. We observe a clean trade-off between validation and test accuracy with respect to α, but the test accuracy error bars are larger than those of the model that includes $\mathbf{z}_s$.

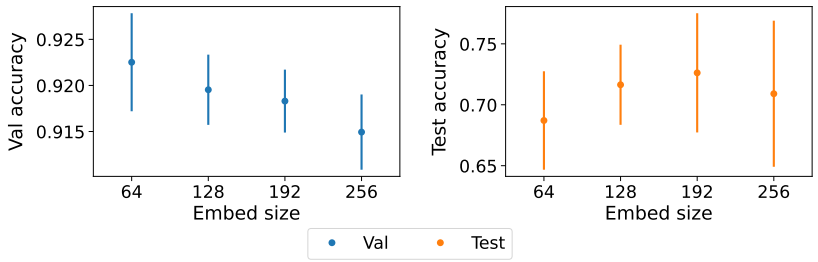


Figure 14: Camelyon17-WILDS embedding size ($D_{\mathbf{z}_c}$ and $D_{\mathbf{z}_s}$) ablation study for SCBD with DenseNet-121 encoders and $\alpha = 192$. The results are relatively consistent across different embedding sizes.

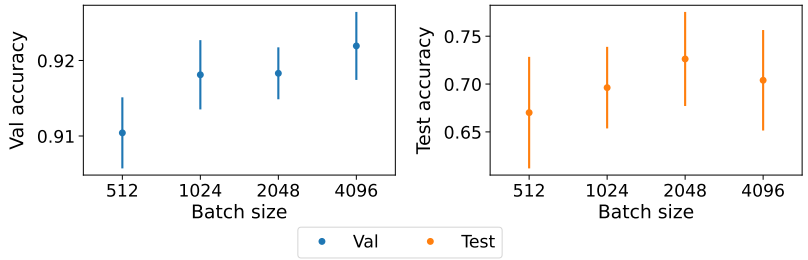


Figure 15: Camelyon17-WILDS batch size ablation study for SCBD with DenseNet-121 encoders and $\alpha = 192$. The results are generally better for larger batch sizes, which was also observed by the authors of SCL (Khosla et al., 2020).

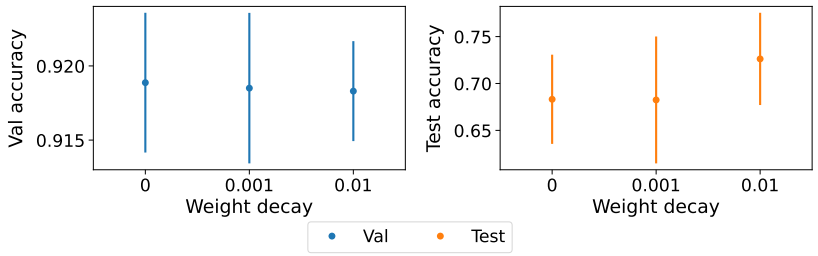
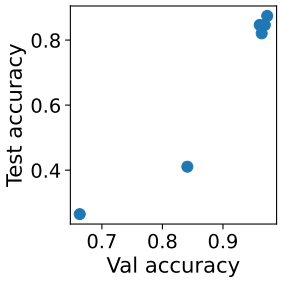
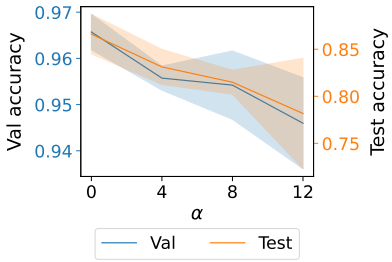


Figure 16: Camelyon17-WILDS weight decay ablation study for SCBD with DenseNet-121 encoders and $\alpha = 192$. The results are relatively consistent across different degrees of weight decay.

A.2.3 PACS

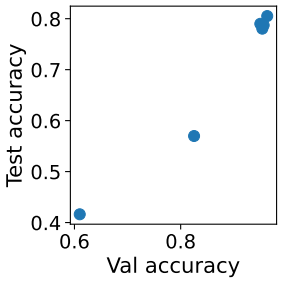


(a) In- and out-of-distribution performance are positively correlated, which violates the assumptions made by SCBD.

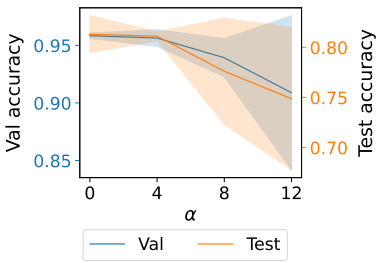


(b) Due to the violation of assumptions made by SCBD, increasing α does not lead to a trade-off between in- and out-of-distribution performance.

Figure 17: PACS with art painting as the test domain.

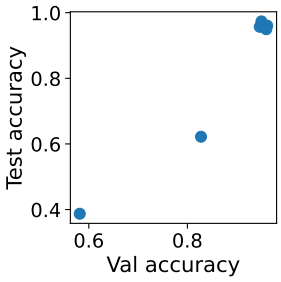


(a) In- and out-of-distribution performance are positively correlated, which violates the assumptions made by SCBD.

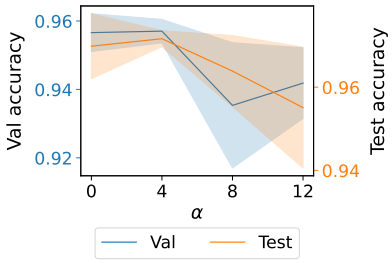


(b) Due to the violation of assumptions made by SCBD, increasing α does not lead to a trade-off between in- and out-of-distribution performance.

Figure 18: PACS with cartoon as the test domain.

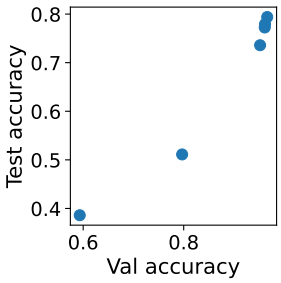


(a) In- and out-of-distribution performance are positively correlated, which violates the assumptions made by SCBD.

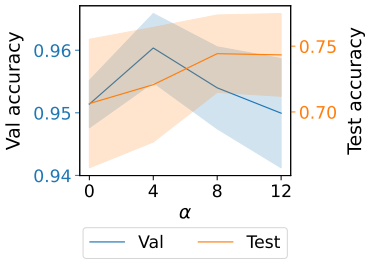


(b) Due to the violation of assumptions made by SCBD, increasing α does not lead to a trade-off between in- and out-of-distribution performance.

Figure 19: PACS with photo as the test domain.



(a) In- and out-of-distribution performance are positively correlated, which violates the assumptions made by SCBD.



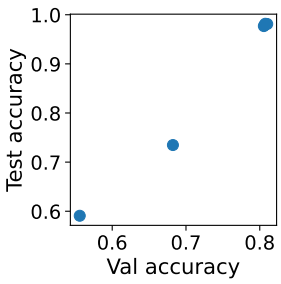
(b) Due to the violation of assumptions made by SCBD, increasing α does not lead to a trade-off between in- and out-of-distribution performance.

Figure 20: PACS with sketch as the test domain.

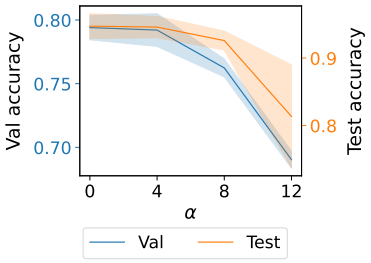
Table 6: Test accuracy (%) for PACS with three random seeds.

Algorithm	Art painting	Cartoon	Photo	Sketch	Average
SCBD ($\alpha = 0$)	86.6 ± 2.1	81.3 ± 1.9	97.0 ± 0.8	70.6 ± 4.9	83.9
ERM	88.1 ± 0.1	77.9 ± 1.3	97.8 ± 0.0	79.1 ± 0.9	85.7
CORAL	87.7 ± 0.6	79.2 ± 1.1	97.6 ± 0.0	79.4 ± 0.7	86.0
DANN	85.9 ± 0.5	79.9 ± 1.4	97.6 ± 0.2	75.2 ± 2.8	84.6
IRM	85.0 ± 1.6	77.6 ± 0.9	96.7 ± 0.3	78.5 ± 2.6	84.4
Group DRO	86.4 ± 0.3	79.9 ± 0.8	98.0 ± 0.3	72.1 ± 0.7	84.1

A.2.4 VLCS

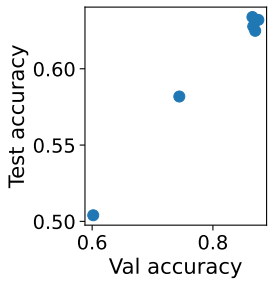


(a) In- and out-of-distribution performance are positively correlated, which violates the assumptions made by SCBD.

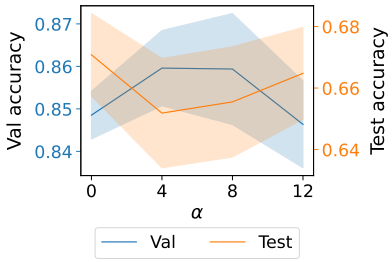


(b) Due to the violation of assumptions made by SCBD, increasing α does not lead to a trade-off between in- and out-of-distribution performance.

Figure 21: VLCS with Caltech101 as the test domain.

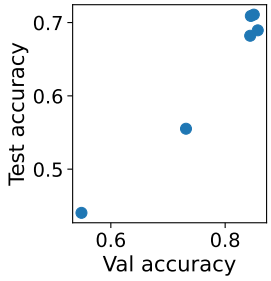


(a) In- and out-of-distribution performance are positively correlated, which violates the assumptions made by SCBD.

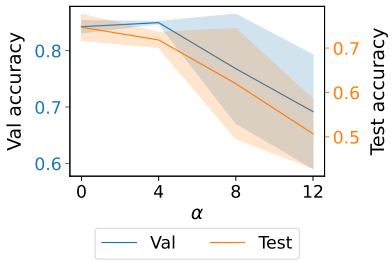


(b) Due to the violation of assumptions made by SCBD, increasing α does not lead to a trade-off between in- and out-of-distribution performance.

Figure 22: VLCS with LabelMe as the test domain.

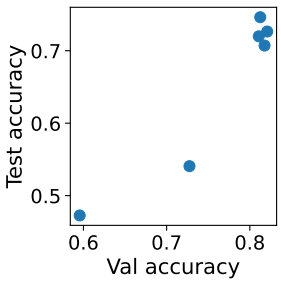


(a) In- and out-of-distribution performance are positively correlated, which violates the assumptions made by SCBD.

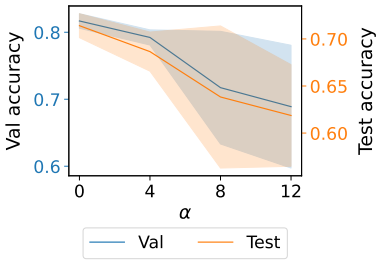


(b) Due to the violation of assumptions made by SCBD, increasing α does not lead to a trade-off between in- and out-of-distribution performance.

Figure 23: VLCS with SUN09 as the test domain.



(a) In- and out-of-distribution performance are positively correlated, which violates the assumptions made by SCBD.



(b) Due to the violation of assumptions made by SCBD, increasing α does not lead to a trade-off between in- and out-of-distribution performance.

Figure 24: VLCS with VOC2007 as the test domain.

Table 7: Test accuracy (%) for VLCS with three random seeds.

Algorithm	Caltech101	LabelMe	SUN09	VOC2007	Average
SCBD ($\alpha = 0$)	94.7 ± 1.8	67.1 ± 1.3	74.7 ± 3.0	71.4 ± 1.3	77.0
ERM	97.6 ± 1.0	63.3 ± 0.9	72.2 ± 0.5	76.4 ± 1.5	77.4
CORAL	98.8 ± 0.1	64.6 ± 0.8	71.7 ± 1.4	75.8 ± 0.4	77.7
DANN	98.5 ± 0.2	64.9 ± 1.1	73.1 ± 0.7	78.3 ± 0.3	78.7
IRM	97.6 ± 0.3	65.0 ± 0.9	72.2 ± 0.5	76.4 ± 1.5	78.1
Group DRO	97.7 ± 0.4	62.5 ± 1.1	70.1 ± 0.7	78.4 ± 0.9	77.2

A.3 Batch correction

Batch prediction We begin by computing the perturbation embedding for every example in the dataset, including the training, validation, and test sets. We then discard all examples for which we do not have a corresponding CellProfiler embedding. Using a randomly sampled 60% of the data as the training set, and the remaining data as the test set, we apply logistic regression to predict e given the embeddings, and report the F1 score on the test set.

CORUM prediction Our CORUM prediction task mirrors that of (Wang et al., 2023), with some modifications to ensure a fair comparison with CellProfiler. We begin by computing $\mathbf{z}_c$ for every single-cell image in the dataset, including the training, validation, and test sets. Then, we discard all embeddings for which we do not have a corresponding CellProfiler embedding. The median number of cells per gene is $6{,}000$, and we want to average them to obtain a single embedding per gene. We have four sgRNA sequences for each perturbed gene, and 250 sgRNA sequences for the non-targeting control. We first average the $\mathbf{z}_c$'s across cells for each sgRNA sequence, and then average the resulting sgRNA embeddings that correspond to the same gene. For each gene embedding, we subtract the non-targeting control embedding, then standardize such that each of the 64 components has mean zero and unit variance.

Then, we incorporate the CORUM database, which defines the pairs of genes that belong to the same protein complex. We discard all gene embeddings that are not in this database. We compute the cosine similarity between each pair of gene embeddings, and interpret it as the prediction that they belong to the same family. The prediction target is one if they belong to the same family according to the CORUM database, and zero otherwise. We turn the cosine similarities into binary predictions by across various prediction thresholds by using the i'th percentile as the upper threshold and the $100 - i$'th percentile as the lower threshold for each integer $i \in \{80, \ldots, 100\}$. Finally, we use the binary predictions and prediction targets to obtain a precision and recall at each value of i, and plot the precision and recall curve.

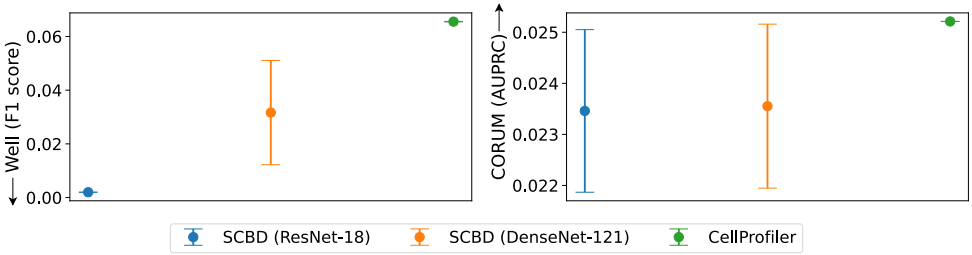


Figure 25: The left subfigure shows the performance of predicting the well label e, while the right subfigure represents the biological content. SCBD with DenseNet-121 encoders are less predictive of the well than CellProfiler, while retaining a similar level of biological information. However, the DenseNet-121 models are more sensitive to batch effects compared to the ResNet-18 models.

B CODE

Here is our implementation of the supervised contrastive and invariance losses.

```
def supcon_loss(z, u, temperature):
    batch_size = len(z)
    u_col = u.unsqueeze(1)
    u_row = u.unsqueeze(0)
    mask_pos = (u_col == u_row).float()
    offdiag_mask = 1. - torch.eye(batch_size)
    mask_pos = mask_pos * offdiag_mask
    logits = torch.matmul(z, z.T) / temperature
    p = mask_pos / mask_pos.sum(dim=1, keepdim=True).clamp(min=1.)
    q = F.log_softmax(logits, dim=1)
    return F.cross_entropy(q, p)

def invariance_loss(zc, e, temperature):
    batch_size = len(zc)
    e_col = e.unsqueeze(1)
    e_row = e.unsqueeze(0)
    mask_pos = (e_col == e_row).float()
    mask_neg = 1. - mask_pos
    offdiag_mask = 1. - torch.eye(batch_size)
    mask_pos = mask_pos * offdiag_mask
    logits = torch.matmul(zc, zc.T) / temperature
    q = F.log_softmax(logits, dim=1)
    log_prob_pos = (q * mask_pos).mean(dim=1)
    log_prob_neg = (q * mask_neg).mean(dim=1)
    return (log_prob_pos - log_prob_neg).abs().mean()
```