Correct-N-Contrast: A Contrastive Approach for Improving Robustness to Spurious Correlations

Michael Zhang [1] **Nimit S. Sohoni** [1] **Hongyang R. Zhang** [2] **Chelsea Finn** [1] **Christopher Ré** [1]

Abstract

Spurious correlations pose a major challenge for robust machine learning. Models trained with empirical risk minimization (ERM) may learn to rely on correlations between class labels and spurious attributes, leading to poor performance on data groups without these correlations. This is challenging to address when the spurious attribute labels are unavailable. To improve worst-group performance on spuriously correlated data without training attribute labels, we propose Correct-N-Contrast (CNC), a contrastive approach to directly learn representations robust to spurious correlations. As ERM models can be good spurious attribute predictors, CNC works by (1) using a trained ERM model's outputs to identify samples with the same class but dissimilar spurious features, and (2) training a robust model with contrastive learning to learn similar representations for these samples. To support CNC, we introduce new connections between worst-group error and a representation *alignment loss* that CNC aims to minimize. We empirically observe that worst-group error closely tracks with alignment loss, and prove that the alignment loss over a class helps upper-bound the class's worst-group vs. average error gap. On popular benchmarks, CNC reduces alignment loss drastically, and achieves state-of-the-art worst-group accuracy by **3.6**% average absolute lift. CNC is also competitive with oracle methods that require group labels.

1. Introduction

For many tasks, deep neural networks (NNs) are negatively affected by spurious correlations—dependencies between observed features and class labels that only hold for certain groups of data. For example, consider classifying cows versus camels in natural images. 90% of cows may appear on grass, and 90% of camels on sand. A model trained with standard ERM may learn to minimize average training error by relying on the spurious *background* attribute instead of the desired *animal*, performing well on average yet obtaining high *worst-group* test error (e.g., misclassifying cows on sand as camels) (Ribeiro et al., 2016; Beery et al., 2018). This illustrates a widespread issue where models are systematically biased due to the presence of spurious attributes. This issue is critical to address in fields ranging from algorithmic fairness to medical ML (Blodgett et al., 2016; Buolamwini & Gebru, 2018; Hashimoto et al., 2018).

How can we improve robustness to spurious correlations and reduce worst-group error? If group or spurious attribute labels are available, one option is to reweight groups during training to directly minimize worst-group error, e.g. via group distributionally robust optimization (GDRO) (Sagawa et al., 2019). However, such group labels may be expensive to obtain or unknown *a priori* (Oakden-Rayner et al., 2020), limiting the applicability of these "oracle" methods. To tackle spurious correlations without these labels, many prior works recognize that because ERM models can learn spurious correlations, such models can help infer the groups or spurious attributes. These methods thus first infer data groups with a trained ERM model, before using these inferred groups to train a more robust model. For example, Sohoni et al. (2020) cluster an ERM model's feature representations to infer groups, before running GDRO using these clusters as groups. Nam et al. (2020); Liu et al. (2021) treat an ERM model's misclassifications as minority group samples, and upweight these groups to train a robust model. Ahmed et al. (2021); Creager et al. (2021) use invariance objectives to both infer groups with an ERM model and train a more robust model. However, while these methods significantly improve worst-group error over ERM without training group labels, they still perform worse than methods that use group labels (e.g., GDRO).

In this work, we thus aim to further improve worst-group accuracy without requiring group labels. We start with the motivating observation that across spurious correlation settings, a neural network's worst-group accuracy strongly

[1]Stanford University. Stanford, CA, USA [2]Northeastern University. Boston, MA, USA. Correspondence to: Michael Zhang <mzhang@cs.stanford.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

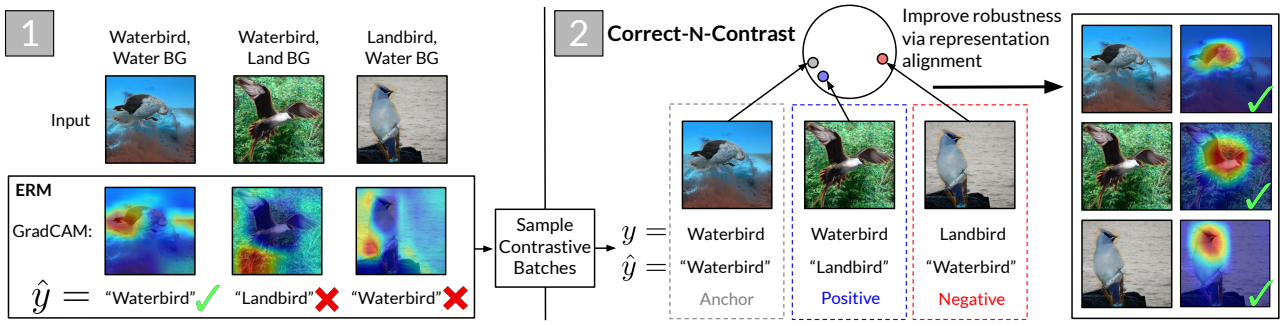


Figure 1. (1) ERM-trained models classify by spurious features, shown via GradCAM (Selvaraju et al., 2017). (2) CNC learns similar representations for same-class samples with different ERM predictions to ignore spurious attributes and classify samples correctly.

tracks how well its representations—i.e., the outputs from its last hidden layer—exhibit dependence *only* on ground-truth labels, and *not* on spurious attributes. We quantify this property via estimated mutual information as well as a notion of geometric representation *alignment*, inspired by prior work in contrastive learning (Wang & Isola, 2020). Here, the alignment measures how close samples with the same class but different spurious attributes embed in representation space. We first empirically observe that this dependence consistently holds across datasets with increasingly strong spurious correlations, and also helps explain when upweighting methods (e.g., JTT (Liu et al., 2021)) improve worst-group error over ERM. We then theoretically show that a model's alignment loss for a class can be used to upper-bound its worst-group versus average error gap for that class. Thus, by improving alignment while keeping average error low for a class, we can help improve the class's worst-group error. However, current methods do not directly optimize for representation-level properties, suggesting one underexplored direction to improve worst-group error.

We thus propose Correct-N-Contrast (CNC), a two-stage contrastive learning approach for better-aligned representations and improved robustness to spurious correlations. The key idea is to use contrastive learning to "push together" representations for samples with the same class but different spurious attributes, while "pulling apart" those with different classes and the same spurious attribute (Fig. 1). CNC thus improves intra-class alignment while also maintaining inter-class separability, encouraging models to rely on features predictive of class labels but not spurious attributes. As we do not know the spurious attribute labels, CNC first infers the spurious attributes using a regularized ERM model trained to predict class labels (similar to prior work). Different from these works, we use these predictions to train another robust model via contrastive learning and a novel sampling strategy. We randomly sample an anchor, select samples with the same class but different ERM predictions as hard positives to "push together," and select samples from different classes but the same ERM prediction as hard negatives to "pull apart." This encourages ignoring spurious differences and learning class-specific similarities between anchor and positives, and conversely ignoring spurious similarities and learning class-specific differences between anchor and negatives. CNC thus corrects for the ERM model's learned spurious correlations via contrastive learning.

We evaluate CNC on four spurious correlation benchmarks. Among methods that do not assume training group labels, CNC substantially improves worst-group accuracy, obtaining up to **7.7%** absolute lift (from 81.1% to 88.8% on CelebA), and averaging **3.6%** absolute lift over the second-best method averaged over all tasks (JTT, Liu et al. (2021)). CNC also nearly closes the worst-group accuracy gap with methods requiring training group labels, only falling short of GDRO's worst-group accuracy by 0.9 points on average. To help explain this lift, we find that CNC indeed improves alignment and learns representations with substantially higher dependence on classes over spurious attributes. Finally, we run additional ablation experiments that show that: CNC is more robust to noisy ERM predictions than prior methods; with spurious attribute labels, CNC improves worst-group accuracy over GDRO by **0.9%** absolute on average; and CNC's sampling approach improves performance compared to alternative approaches. These results show that our contrastive learning and sampling strategies are effective techniques to tackle spurious correlations.

Summary. We summarize our contributions as follows: (1) We empirically show that worst-group error correlates with alignment loss, and theoretically analyze this connection. (2) We propose CNC, a two-stage contrastive approach with a hard negative sampling scheme to improve representation alignment and thus train models more robust to spurious correlations. (3) We show that CNC achieves state-of-the-art worst-group accuracy on three benchmarks and learns better-aligned representations less reliant on spurious features.

2. Preliminaries

Problem setup. We present our setting and the loss objectives following Sagawa et al. (2019). Let $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_n\}$ be our training dataset of size n. Each datapoint has an observed feature vector $x_i \in \mathcal{X}$, label

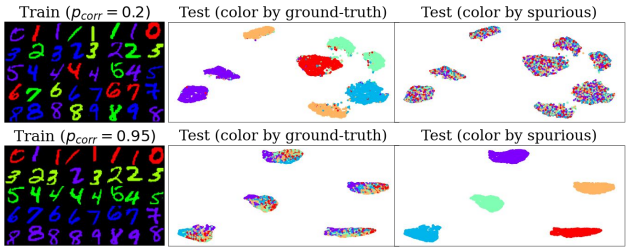


Figure 2. UMAPs (Uniform Manifold Approximation and Projection) (McInnes et al., 2018) for dimensionality reduction and visualization of ERM-trained hidden-layer representations. Models trained on spuriously correlated data ($p_{\text{corr}} = 0.95$) exhibit greater dependence on spurious attributes over ground-truth concepts versus models trained on spuriously uncorrelated data ($p_{\text{corr}} = 0.20$).

$y_i \in \mathcal{Y}$, and *unobserved* spurious attribute $a_i \in \mathcal{A}$. The set of groups $\mathcal{G}$ is defined as the set of all combinations of class label and spurious attribute pairs, i.e. $\mathcal{G} = \mathcal{Y} \times \mathcal{A}$. Let $C = |\mathcal{Y}|$ be the number of classes and $K = |\mathcal{G}|$ be the number of groups. We assume that each example (x_i, y_i, a_i) is drawn from an unknown joint distribution P, and at least one sample from each group is observed in the training data. Let P_g be the distribution conditioned on $(y, a) = g$, for any $g \in \mathcal{G}$. Given a model $f_\theta : \mathcal{X} \mapsto \mathbb{R}^C$ and a convex loss $\ell : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$, the *worst-group* loss is:

$$\mathcal{L}_{\text{wg}}(f_\theta) := \max_{g \in \mathcal{G}} \mathbb{E}_{(x,y,a)\sim P_g}[\ell(f_\theta(x), y)]. \quad (1)$$

ERM minimizes the training loss as a surrogate for the expected average population loss $\mathcal{L}_{\text{avg}}$:

$$\mathcal{L}_{\text{avg}}(f_\theta) := \mathbb{E}_{(x,y,a)\sim P}[\ell(f_\theta(x), y)] \quad (2)$$

While ERM is the standard way to train NNs, spurious correlations can cause ERM to obtain high minority group error even with low average error. Minimizing the empirical version of (1) via GDRO is a strong baseline for improving worst-group error, if training group labels $\{a_1, \ldots, a_n\}$ are available (Sagawa et al., 2019). We tackle the more challenging setting where training group labels are *not available*.

Contrastive learning. We briefly describe contrastive learning (Chen et al., 2020), a central component of our approach. Let f_θ be a neural network model with parameters θ. Let the encoder $f_{\text{enc}} : \mathcal{X} \mapsto \mathbb{R}^d$ be the feature representation layers of f_θ. Let $f_{\text{cls}} : \mathbb{R}^d \mapsto \mathbb{R}^C$ be the classification layer of f_θ, which maps encoder representations to one-hot label vectors. We learn f_{enc} with the *supervised contrastive loss* $\mathcal{L}^{\text{sup}}_{\text{con}}$ proposed in Khosla et al. (2020). For each anchor x, we sample M positives $\{x_i^+\}_{i=1}^M$ and N negatives $\{x_i^-\}_{i=1}^N$. Let $y, \{y_i^+\}_{i=1}^M, \{y_i^-\}_{i=1}^N$ be the labels and $z, \{z_i^+\}_{i=1}^M, \{z_i^-\}_{i=1}^N$ be the normalized outputs of $f_{\text{enc}}(x)$ for the anchor, positives, and negatives respectively. With input x mapped to z, the training objective for the encoder is to minimize $\mathcal{L}^{\text{sup}}_{\text{con}}(x; f_{\text{enc}})$, defined as

$$\mathbb{E}\left[-\log \frac{\exp(z^\top z_i^+/\tau)}{\sum_{m=1}^M \exp(z^\top z_m^+/\tau) + \sum_{n=1}^N \exp(z^\top z_n^-/\tau)}\right], \quad (3)$$

where $\tau > 0$ is a scalar temperature hyperparameter and the expectation is over $z, \{z_i^+\}, \{z_i^-\}$. Minimizing Eq. 3 leads to z being closer to z^+ than z^- in representation space.

3. The impact of spurious correlations on learned data representations

We present our key observation that a model's worst-group accuracy correlates with how well its learned representations depend on the class labels, and *not* the spurious attributes. We draw connections between a neural network's worst-group error and its alignment and mutual information metrics, noting a strong inverse relationship between worst-group accuracy and a class-specific alignment loss, and then theoretically justify this relationship.

3.1. Understanding worst-group performance using representation metrics

We first illustrate that when neural networks are trained with standard ERM on spuriously correlated data, their hidden layer representations exhibit high dependence on the spurious attribute. To better understand and connect this behavior to worst-group error, we quantify these results using representation alignment (cf. Eq. 4) and mutual information metrics. We observe that these metrics explain trends in ERM's worst-group accuracy on various spuriously correlated datasets. These trends also apply to upsampling methods that mitigate the impact of spurious features.

Example setup. We model spurious correlations with CMNIST*, a colored MNIST dataset inspired by Arjovsky et al. (2019). There are 5 digit classes and 5 colors. We color a fraction p_{corr} of the training samples with a color a associated with each class y, and color the test samples uniformly-randomly. To analyze learned representations, we train a LeNet-5 CNN (LeCun et al., 1989) with ERM to predict digit classes, and inspect the outputs of the last hidden layer $z = f_{\text{enc}}(x)$. As shown in Fig. 2, with low p_{corr}, models learn representations with high dependence on the actual digit classes. However, with high p_{corr}, we learn z highly dependent on a, despite only training to predict y.

Representation metrics. To quantify this behavior, we use two metrics designed to capture how well the learned representations exhibit dependence on the class label versus the spurious attributes. First, we compute an *alignment loss* $\hat{\mathcal{L}}_{\text{align}}(f_{\text{enc}}; g, g')$ between two groups $g = (y, a)$ and $g' = (y, a')$ where $a \neq a'$. This measures how well f_{enc} maps samples with the same class, but different spurious attributes, to nearby vectors via Euclidean distance.

Letting G and G' be the subsets of training data in groups g and g' respectively, we define $\hat{\mathcal{L}}_{\text{align}}(f_{\text{enc}}; g, g')$ as

$$\frac{1}{|G|}\frac{1}{|G'|} \sum_{(x,y,a)\in G} \sum_{(x',y,a')\in G'} \|f_{\text{enc}}(x) - f_{\text{enc}}(x')\|_2 \quad (4)$$

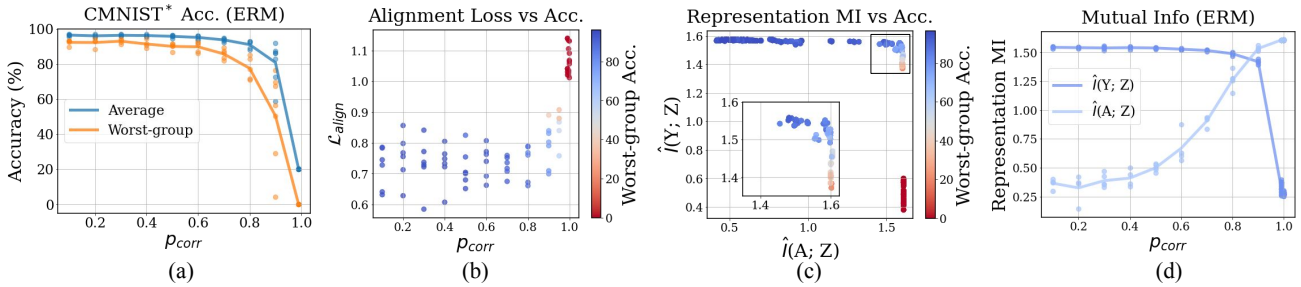


Figure 3. Accuracy and representation metrics from ERM models trained on increasingly spuriously correlated Colored MNIST. Lower worst-group accuracy (Fig. 3a) corresponds to both higher alignment loss (Fig. 3b) and $\hat{I}(Y;Z) < \hat{I}(A;Z)$ (Fig. 3c, Fig. 3d).

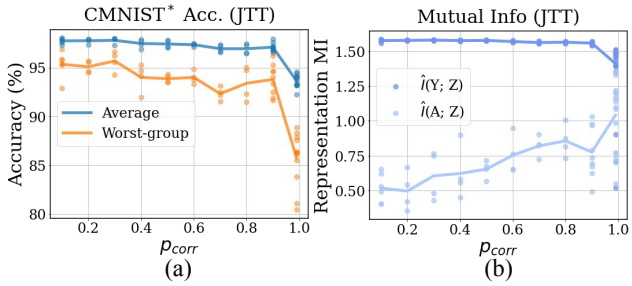


Figure 4. Higher worst-group accuracy with JTT (versus Fig. 3a) coincides with keeping $\hat{I}(Y;Z) \gg \hat{I}(A;Z)$.

Thus, lower $\hat{\mathcal{L}}_{\text{align}}$ means better alignment. We also quantify representation dependence by estimating the mutual information (MI) of a model's learned representations with the class label, i.e. $\hat{I}(Y;Z)$ and the spurious attributes $\hat{I}(A;Z)$. We defer computational details to Appendix E.

Results for ERM. In Fig. 3 we find that worst-group error is strongly associated with both alignment and mutual information (MI) metrics. As p_{corr} increases, ERM models not only drop in worst-group accuracy, but also incur higher alignment loss (Fig. 3ab). Fig. 3c further illustrates this with MI. We plot the estimated MI and worst-group accuracy for models at each epoch. A substantial drop in worst-group accuracy occurs with high $\hat{I}(A;Z)$ (especially when $\hat{I}(A;Z) > \hat{I}(Y;Z)$, even when $\hat{I}(Y;Z)$ is high). Fig. 3d also captures this trend: as p_{corr} increases, $\hat{I}(A;Z)$ does as well while $\hat{I}(Y;Z)$ decreases.

Results for JTT. In Fig. 4, we also show that this relation holds when training with another recent (upsampling-based) approach, JTT (Liu et al., 2021). With high p_{corr}, JTT achieves higher worst-group accuracy compared to ERM, and this corresponds to learning representations with high $\hat{I}(Y;Z)$ and low $\hat{I}(A;Z)$. However, we note that JTT and other previous approaches do not explicitly optimize representation-level metrics, suggesting a new direction to improve worst-group performance.

3.2. Justification that better alignment encourages lower worst-group loss

Next, we give a rigorous justification of the relation from lower alignment loss to lower worst-group loss. For any label $y \in \mathcal{Y}$, let $\mathcal{G}_y$ be the set of groups with label y in $\mathcal{G}$. Let $\mathcal{L}_{\text{wg}}(f_\theta; y)$ be the worst-group loss among groups in $\mathcal{G}_y$:

$$\mathcal{L}_{\text{wg}}(f_\theta; y) := \max_{g \in \mathcal{G}_y} \mathop{\mathbb{E}}_{(x,\tilde{y},a)\sim P_g} [\ell(f_\theta(x), \tilde{y})] .$$

Let $\mathcal{L}_{\text{avg}}(f_\theta; y)$ be the average loss among groups in $\mathcal{G}_y$:

$$\mathcal{L}_{\text{avg}}(f_\theta; y) := \mathop{\mathbb{E}}_{(x,\tilde{y},a)\sim P:\forall a \in \mathcal{A}} [\ell(f_\theta(x), \tilde{y})] .$$

Additionally, let $\hat{\mathcal{L}}_{\text{align}}(f_\theta; y)$ be the largest cross-group alignment loss among groups in $\mathcal{G}_y$:

$$\hat{\mathcal{L}}_{\text{align}}(f_\theta; y) := \max_{g \in \mathcal{G}_y, g' \in \mathcal{G}_y : g \neq g'} \hat{\mathcal{L}}_{\text{align}}(f_{\text{enc}}; g, g'). \quad (5)$$

We state our result as follows.

Theorem 3.1. *In the setting described above, suppose the weight matrix of the linear classification layer W satisfies $\|W\|_2 \leq B$, for some $B > 0$. Suppose the loss function $\ell(x, y)$ is C_1-Lipschitz in x and bounded from above by C_2, for some $C_1 >$ and $C_2 > 0$. Let n_g be the size of any group $g \in \mathcal{G}$ in the training set. Then, for any $\delta > 0$, with probability $1 - \delta$, the following holds for any $y \in \mathcal{Y}$:*

$$\begin{aligned} &\mathcal{L}_{wg}(f_\theta; y) - \mathcal{L}_{avg}(f_\theta; y) \quad (6) \\ &\leq BC_1 \cdot \hat{\mathcal{L}}_{align}(f_\theta; y) + \max_{g \in \mathcal{G}_y} C_2 \sqrt{8 \log(|\mathcal{G}_y|/\delta)/n_g}. \end{aligned}$$

The broader implication of our result is that reducing the alignment loss closes the gap between worst-group and average-group losses. The proof is deferred to Appendix B.

Broader implications. We summarize this section with two takeaways: **(1)** When trained on spuriously correlated datasets, ERM networks learn data representations highly dependent on spurious attributes. Representation clusters (Sohoni et al., 2020) or the ERM model's outputs (Liu et al., 2021; Nam et al., 2020) can thus serve as (noisy) pseudolabels for spurious attributes. **(2)** Both representation metrics correlate with worst-group error, such that a viable way to improve worst-group error is to improve alignment within each class. Both takeaways motivate our approach next.

4. Our approach: Correct-N-Contrast (CNC)

We now present CNC, a two-stage method to improve worst-group performance and robustness to spurious correlations,

without training group labels. Similar to prior works, our first stage trains an ERM model with proper regularization[1] on the training set, in order to infer spurious attributes.

The key difference is our second stage: we aim to train a more robust model by learning representations such that samples in the same class but different groups are close to each other. To do so, we use a contrastive learning strategy, proposing a new sampling scheme where we treat samples with the same class but different spurious attributes as distinct "views" of the same class (anchors and positives), while sampling anchors and negatives as datapoints with the same inferred spurious attributes but different classes. By training the second stage model with contrastive learning over these samples, we intuitively encourage the second model to "pull together" samples' representations based on shared class-specific features, while ignoring different spurious features. To obtain such samples, we use the initial ERM model's predictions as spurious attribute proxies.

Later in Sec. 5.1 and Sec. 5.2, we show that CNC indeed reduces $\hat{\mathcal{L}}_{\text{align}}(f_\theta; y)$ and substantially improves worst-group accuracy. In Sec. 5.3 we also show that alternative sampling strategies degrade performance. We include further details on both stages below, and summarize CNC in Algorithm 1.

Stage 1: Inferring pseudo group labels. We train an initial model $f_{\hat{\theta}}$ on the training dataset $\{(x_i, y_i)\}_{i=1}^n$ with ERM and regularization, and save its predictions $\{\hat{y}_i\}_{i=1}^n$ on the training datapoints. We consider two ways to get predictions $\hat{y}$: standard argmax over the ERM model's final-layer outputs (as in Liu et al. (2021)), and clustering its last hidden-layer outputs into C clusters[2] (similar to Sohoni et al. (2020)). While both approaches exploit the ERM model's learned spurious correlations, we find clustering to lead to better performance (cf. Appendix E.2).

Stage 2: Supervised contrastive learning. Next, we train a robust model with supervised contrastive learning using the ERM predictions. Our approach is based on standard contrastive learning methods (Chen et al., 2020; Khosla et al., 2020), but we introduce new "contrastive batch" sampling and optimization objectives in order to induce robustness to spurious correlations.

Contrastive batch sampling. As described in Sec. 2, contrastive learning involves sampling anchors, positives, and negatives with the general form $\{x\}, \{x^+\}, \{x^-\}$. Here, we wish to sample points such that by maximizing the similarity between anchors and positives (and keeping anchors and negatives apart), the Stage 2 model "ignores" spurious similarities while learning class-consistent dependencies. For each batch we randomly sample an anchor $x_i \in X$ with label y_i and ERM prediction $\hat{y}_i = y$, M positives with the same class as y_i but a different ERM model prediction than $\hat{y}_i$, and N negatives with different classes than y_i but the same ERM model prediction as $\hat{y}_i$. For more comparisons per batch, we also switch anchor and positive roles. We include implementation details in Appendix A.2.

Algorithm 1 Correct-N-Contrast (CNC)

Input: Training dataset (X, Y); # positives M; # negatives N; learning rate η, # epochs K.
Stage 1: Inferring pseudo group labels
Train ERM model $f_{\hat{\theta}}$ on (X, Y); save $\hat{y}_i := f_{\hat{\theta}}(x_i)$.
Stage 2: Supervised contrastive learning
Initialize "robust" model f_θ (e.g., with random weights)
for epoch $1, \ldots, K$ **do**
  for anchor $(x, y) \in \{(X, Y) : \hat{y} = y\}$ **do**
    (Let $\hat{y} := f_{\hat{\theta}}(x)$ be the ERM model prediction of x.)
    Get M positives $\{(x_m^+, y_m^+)\}$ where $y_m^+ = y$, $\hat{y}_m^+ \neq \hat{y}$.
    Get N negatives $\{(x_n^-, y_n^-)\}$ where $y_n^- \neq y$, $\hat{y}_n^- = \hat{y}$.
    Update f_θ by $\theta \leftarrow \theta - \eta \cdot \nabla \hat{\mathcal{L}}(f_\theta; x, y)$ (cf. Eq. (7)) with anchor, M positives, and N negatives.
return final model f_θ from Stage 2.

Optimization objective and updating procedure. Recall that we seek to learn aligned representations to improve robustness to spurious correlations. Thus, we also jointly train the full model to classify datapoints correctly. As we have the training *class* labels, we jointly update the model's encoder layers f_{enc} with a contrastive loss, and the full model f_θ with a cross-entropy loss. Our overall objective is:

$$\hat{\mathcal{L}}(f_\theta; x, y) = \lambda \hat{\mathcal{L}}_{\text{con}}^{\text{sup}}(f_{\text{enc}}; x, y) + (1 - \lambda)\hat{\mathcal{L}}_{\text{cross}}(f_\theta; x, y). \quad (7)$$

Here $\hat{\mathcal{L}}_{\text{con}}^{\text{sup}}(f_{\text{enc}}; x, y)$ is the supervised contrastive loss of x and its positive and negative samples, similar to Eq. (3) (see Eq (8), Appendix A.2 for the full equation); $\hat{\mathcal{L}}_{\text{cross}}(f_\theta; x, y)$ is average cross-entropy loss over x, the M positives, and N negatives; $\lambda \in [0, 1]$ is a balancing hyperparameter.

To calculate the loss, we first forward propagate one batch $\left(x_i, \{x_m^+\}_{m=1}^M, \{x_q^-\}_{q=1}^N\right)$ through f_{enc} and normalize the outputs to obtain representation vectors $\left(z_i, \{z_m^+\}_{m=1}^M, \{z_q^-\}_{q=1}^N\right)$. To learn closely aligned z_i and z^+ for all $\{z_m^+\}_{m=1}^M$, we update f_{enc} with the $\hat{\mathcal{L}}_{\text{out}}^{\text{sup}}(\cdot; f_{\text{enc}})$ loss. Finally, we also pass the unnormalized encoder outputs f_{enc} to the classifier layers f_{cls} and compute a batch-wise cross-entropy loss $\hat{\mathcal{L}}_{\text{cross}}(f_\theta)$ using each sample's class labels and f_θ's outputs. Further details are in Appendix A.

5. Experimental results

We conduct experiments to answer the following questions: (1) Does CNC improve worst-group performance over prior state-of-the-art methods on datasets with spurious correlations? (2) To help explain any improvements, do CNC-learned representations actually exhibit greater alignment

[1] As we train on the same dataset we infer spurious attributes on, regularization (via high weight decay or early stopping) is to prevent the ERM model from memorizing train labels. This is standard (e.g., (Sohoni et al., 2020; Liu et al., 2021)). In Sec. 5.3 we show that we do not require extremely accurate spurious attribute predictions to substantially improve robustness in practice.

[2] Recall that C is the number of classes.

Table 1. Worst-group and average accuracies. **1st** / 2nd best worst-group accuracies **bolded** / underlined. On image datasets, CNC substantially improves worst-group accuracy over comparable methods without group labels, competing with GDRO. CNC also competes with SoTA on CivilComments. Starred results from original papers. Others run over 3 seeds. More implementation details in Appendix E.

	CMNIST*		Waterbirds		CelebA		CivilComments-WILDS	
Accuracy (%)	Worst-group	Average	Worst-group	Average	Worst-group	Average	Worst-group	Average
ERM	0.0 (0.0)	20.1 (0.2)	62.6 (0.3)	97.3 (1.0)	47.7 (2.1)	94.9 (0.3)	58.6 (1.7)	92.1 (0.4)
LfF	0.0 (0.0)	25.0 (0.5)	78.0 (-)*	91.2*	77.2 (-)*	85.1 (-)*	58.8 (-)*	92.5 (-)*
GEORGE	76.4 (2.3)	89.5 (0.3)	83.8 (1.0)*	93.9 (0.8)*	54.9 (1.9)*	94.5 (0.2)*	-	-
PGI	73.5 (1.8)	88.5 (1.4)	79.5 (1.9)	95.5 (0.8)	85.3 (0.3)	87.3 (0.1)	-	-
CIM	0.0 (0.0)	36.8 (1.3)	77.2 (-)*	95.6 (-)*	83.6 (-)*	90.6 (-) *	N/A	N/A
EIIL	72.8 (6.8)	90.7 (0.9)	77.2 (1.0)	96.5 (0.2)	81.7 (0.8)	85.7 (0.1)	67.0 (2.4)*	90.5 (0.2)*
JTT	74.5 (2.4)	90.2 (0.8)	83.8 (1.2)	89.3 (0.7)	81.5 (1.7)	88.1 (0.3)	**69.3** (-)*	91.1 (-)*
CNC (Ours)	**77.4 (3.0)**	90.9 (0.6)	**88.5 (0.3)**	90.9 (0.1)	**88.8 (0.9)**	89.9 (0.5)	68.9 (2.1)	81.7 (0.5)
Group DRO	78.5 (4.5)	90.6 (0.1)	89.9 (0.6)	92.0 (0.6)	88.9 (1.3)	93.9 (0.1)	69.8 (2.4)	89.0 (0.3)

and class-only dependence, and how is this impacted by the strength of a spurious correlation? (3) To better understand CNC's components and properties, how do ablations on the Stage 1 prediction quality and the Stage 2 contrastive sampling strategy impact CNC in practice? We answer each question in the following sections. In Sec. 5.1, we show that CNC substantially improves worst-group accuracy without group labels, averaging 3.6% points higher than prior state-of-the-art. In Sec. 5.2, we verify that more desirable representation metrics consistently coincide with these improvements. Finally in Sec. 5.3, we find CNC to be more robust to inaccurate Stage 1 predictions than alternatives, show that CNC further improves robustness with group labels (improving worst-group accuracy by 0.9 points over GDRO), and validate the importance of CNC's sampling criteria. We present additional ablations on CNC's components, including the alignment approach, in Appendix C. We briefly describe the benchmark evaluation tasks below. We run CMNIST* with $p_{\text{corr}} = 0.995$. Following prior work (Sagawa et al., 2019), we report all results with early stopping with respect to worst-group validation accuracy. Further details on datasets, models, and hyperparameters are in Appendix E.

Waterbirds (Sagawa et al., 2019): We classify $\mathcal{Y} = \{\text{waterbird}, \text{landbird}\}$. 95% of images have the same bird type ($\mathcal{Y}$) and background type ($\mathcal{A} = \{\text{water}, \text{land}\}$) type.

CelebA (Liu et al., 2015): We classify celebrities' hair color $\mathcal{Y} = \{\text{blond}, \text{not blond}\}$ with $\mathcal{A} = \{\text{male}, \text{female}\}$. Only 6% of blond celebrities in the dataset are male.

CivilComments-WILDS (Borkan et al., 2019; Koh et al., 2021): We classify $\mathcal{Y} = \{\text{toxic}, \text{nontoxic}\}$ comments. $\mathcal{A}$ denotes a mention of one of eight demographic identities.

5.1. Comparison of worst-group performance

To study (1), we evaluate CNC on image classification and NLP tasks with spurious correlations. As baselines, we compare against standard ERM and an 'oracle' GDRO approach that assumes access to the group labels. We also compare against recent methods that tackle spurious correlations without requiring group labels: GEORGE (Sohoni et al., 2020), Learning from Failure (LfF) (Nam et al., 2020), Predictive Group Invariance (PGI) (Ahmed et al., 2021), Environment Inference for Invariant Learning (EIIL) (Creager et al., 2021), Contrastive Input Morphing (CIM) (Taghanaki et al., 2021), and Just Train Twice (JTT) (Liu et al., 2021). In Appendix C.4, we also compare CNC with reported worst-group accuracy results using more recent baselines from Idrissi et al. (2021), who explore subsampling (SUBY) and reweighting (RWY) to balance classes as baselines to improve worst-group accuracy.

Results are in Table 1. CNC achieves **highest** worst-group accuracy among all methods without training group labels on CMNIST*, Waterbirds, and CelebA, and near-SoTA worst-group accuracy on CivilComments-WILDS.

While LfF, GEORGE, PGI, EIIL, and JTT similarly use a trained ERM model to estimate groups, CNC uniquely uses ERM predictions to learn desirable representations via contrastive learning. By contrasting positives and negatives, we reason that CNC more strongly encourages ignoring spurious attributes compared to prior invariance, input transformation, or upweighting approaches. We include additional support for this via GradCAM visualizations in Appendix G.

5.2. Detailed analysis of representation metrics

To shed light on CNC's worst-group accuracy gains, we study if models trained with CNC actually learn representations with higher alignment. Compared to ERM and JTT (which obtained second highest worst-group accuracy on average), CNC learns representations with significantly higher alignment (i.e., lower alignment loss) and lower mutual information with spurious attributes, while having comparable mutual information with class labels (Fig. 5). This corresponds to CNC models achieving the highest worst-group accuracy on Waterbirds and CelebA. Further, while all methods produce representations with high mutual information with class labels (Fig. 5b), compared to other methods, CNC representations drastically reduce mutual information with spurious attributes (Fig. 5c). In Fig. 6, we further illustrate

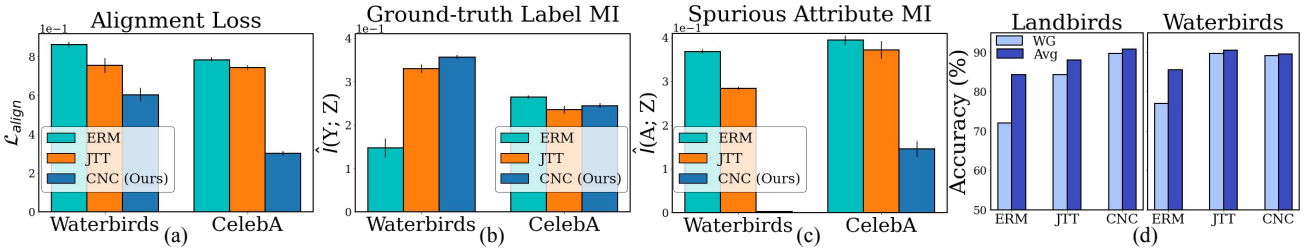


Figure 5. Model alignment loss (a) and mutual information (b, c) after training with ERM, JTT, and CNC. CNC most effectively reduces spurious attribute dependence and obtains smaller gaps for per-class worst-group versus average error (d), as supported by Theorem 3.1.

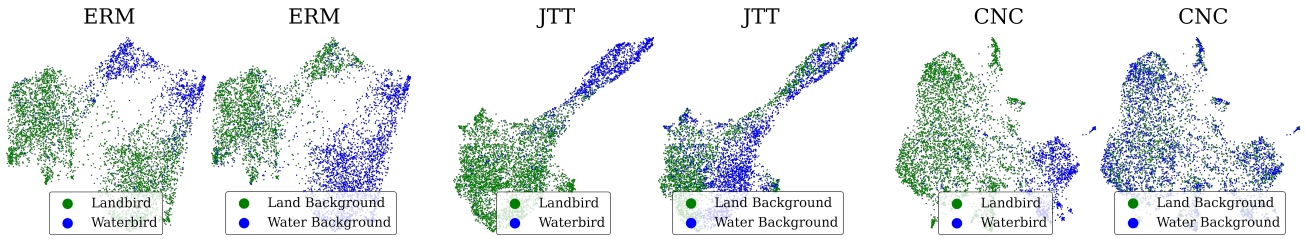


Figure 6. UMAPs of trained Waterbirds representations. ERM representations organize by both ground-truth $\mathcal{Y}$ and spurious $\mathcal{A}$, with greater separability by $\mathcal{A}$. JTT leads to greater separability by $\mathcal{Y}$, but still carries dependence on $\mathcal{A}$. CNC best removes dependence on $\mathcal{A}$.

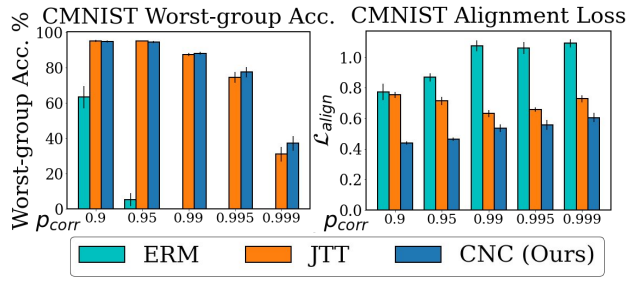


Figure 7. $\mathcal{L}_{\text{align}}$ & WG acc. on CMNIST* with increasing spurious correlation strength. CNC's higher acc. coincides w/lower $\mathcal{L}_{\text{align}}$.

this via UMAP visuals of the learned Waterbirds representations. All methods lead to class-separable representations. However, ERM's representations exhibit stronger separability by spurious attributes, and JTT's also have some spurious attribute dependency. CNC uniquely learns representations that strongly depict class-only dependence.

In addition, to study how this relation between representation metrics and worst-group accuracy scales with the strength of the spurious correlation, we compute representation metrics with CNC, ERM, and JTT models trained on increasingly spurious ($\uparrow p_{\text{corr}}$) CMNIST* datasets (Fig. 7). While CNC and JTT maintain high worst-group accuracy where ERM fails, CNC also performs better in more spurious settings ($p_{\text{corr}} > 0.99$). These improvements are reflected by lower alignment loss (averaged over classes); CNC consistently achieves lowest such loss. We report similar results for mutual information in Appendix C.2.

5.3. Ablation studies

We study the importance of each individual component of our algorithm. We first study how CNC is affected by how well the Stage 1 model predicts spurious attributes, finding CNC more robust to noisy ERM predictions than comparable methods. We next evaluate CNC with true spurious labels, finding CNC to outperform GDRO with this group information. We finally ablate CNC's sampling criteria, and find our hard sampling approach integral for substantial worst-group accuracy gains. In Appendix C.1, we compare other approaches for representation alignment, and find the proposed contrastive approach achieves highest worst-group accuracy.

Table 2. CMNIST* Stage 2 model accuracies with noisy Stage 1 "predictions". CNC is more robust than JTT to increasing noise p.

Noise Probability (p)		0	0.01	0.1	0.25
WG Acc.	JTT	78.4 (2.3)	70.2 (4.7)	53.8 (6.0)	43.6 (7.3)
	CNC	**80.6 (2.8)**	**76.5 (4.1)**	**61.3 (6.0)**	**53.2 (7.0)**
Avg Acc.	JTT	89.7 (0.9)	86.7 (1.3)	79.5 (1.6)	72.9 (1.4)
	CNC	**92.1 (0.5)**	**89.4 (1.4)**	**81.7 (2.0)**	**74.9 (2.0)**

Influence of Stage 1 predictions. ERM predictions can still be noisy proxies for spurious attributes; we thus study if a practical benefit of CNC's contrastive approach is more robustness to this noise compared to alternatives that also use ERM predictions, e.g., JTT's upsampling. We run CNC and JTT with added noise to the same Stage 1 predictions, where JTT upsamples the class-incorrect ones, and compare robust model accuracies. On CMNIST*, starting with true spurious attribute labels as oracle Stage 1 "predictions", we add noise by swapping each label randomly with probability p. In Table 4, while both methods' accuracies drop as p increases (i.e., the "predictions" degrade), CNC consistently achieves higher accuracy that also degrades less than JTT. On real data, to work well CNC also does not require spurious attributes to be perfectly inferred in Stage 1. For Table 1 Waterbirds and CelebA results, the Stage 1 ERM model predicts the spurious attribute with 94.7% and 84.0% accuracy respectively.

Table 3. Accuracy (%) and representation metrics ($\times 10$) for CNC sampling method ablations. Removing components of CNC's sampling criteria reduces worst-group (WG) acc., generally in line with higher $\mathcal{L}_{\text{align}}$, lower class dependence $\hat{I}(Y;Z)$, and higher spurious attribute dependence $\hat{I}(A;Z)$ than default CNC.

	Waterbirds					CelebA				
Method	Avg. Acc.	WG Acc.	$\mathcal{L}_{\text{align}}$ [↓]	$\hat{I}(Y;Z)$ [↑]	$\hat{I}(A;Z)$ [↓]	Avg. Acc.	WG Acc.	$\mathcal{L}_{\text{align}}$ [↓]	$\hat{I}(Y;Z)$ [↑]	$\hat{I}(A;Z)$ [↓]
Neg. by diff. class	95.6 (0.1)	82.2 (1.0)	6.22 (0.12)	**3.59 (0.05)**	1.60 (0.13)	89.5 (0.1)	79.2 (0.4)	3.45 (0.04)	2.34 (0.03)	2.38 (0.06)
Neg. by same pred.	93.6 (0.5)	86.1 (0.5)	6.67 (0.13)	3.50 (0.13)	0.16 (0.09)	88.2 (1.3)	75.0 (5.9)	3.37 (0.59)	2.08 (0.04)	2.13 (0.53)
Pos. by same class	95.8 (0.3)	80.6 (0.7)	6.14 (0.20)	3.50 (0.14)	3.42 (0.16)	87.6 (1.0)	74.4 (0.6)	3.54 (0.46)	2.15 (0.03)	2.78 (0.11)
Pos. by diff. pred.	92.8 (3.0)	86.1 (0.3)	6.66 (0.29)	3.58 (0.07)	0.19 (0.03)	85.8 (0.2)	83.5 (0.3)	3.56 (0.30)	2.16 (0.02)	2.39 (0.20)
SupCon*	**96.8 (0.2)**	62.3 (2.2)	6.93 (0.44)	**3.59 (0.06)**	4.84 (0.13)	**90.4 (0.5)**	61.5 (2.0)	3.83 (0.23)	2.29 (0.04)	3.22 (0.13)
CNC	90.9 (0.1)	**88.5 (0.3)**	**6.02 (0.35)**	3.56 (0.04)	**0.02 (0.01)**	89.9 (0.5)	**88.8 (0.9)**	**3.00 (0.12)**	**2.44 (0.06)**	**1.45 (0.18)**

Table 4. Accuracy (%) using spurious attribute train labels. CNC obtains 0.9% higher worst-group accuracy than GDRO on average.

	CMNIST*		Waterbirds		CelebA	
Acc.	WG	Avg.	WG	Avg.	WG	Avg.
CNC*	**80.6 (2.8)**	**92.4 (0.2)**	**90.1 (0.2)**	**92.4 (0.2)**	**89.2 (1.0)**	92.6 (0.4)
GDRO	78.5 (4.5)	90.6 (0.1)	89.9 (0.6)	92.0 (0.2)	88.9 (1.3)	**93.9 (0.1)**

Training with spurious labels. We next study CNC's performance with true spurious attribute labels on additional datasets. We replace the Stage 1 predictions with true group labels (denoted CNC*), and compare with GDRO–the prior oracle baseline which uses group labels–in Table 4. We find CNC improves with spurious labels, now obtaining 0.9% and 0.2% absolute lift in worst-group and average accuracy over GDRO, suggesting that CNC's contrastive approach can also be beneficial in settings with group labels.

Alternate sampling strategies. We finally study the importance of CNC's sampling by ablating individual criteria. Instead of the default approach that samples negatives from samples with a different class label but same ERM prediction as anchors (CNC), we try sampling negatives only with different classes (Neg. by diff. class), or the same ERM prediction (Neg. by same pred.) as the anchors, keeping the positive sampling approach the same as default. We also ablate the positive sampling approach by keeping the negative sampling the same, and sampling positives from samples only with the same class label (Pos. by same class) or different ERM predictions (Pos. by diff. pred.). We finally try sampling both positives and negatives only by class, similar to Khosla et al. (2020) (SupCon*). Without both hard positive and negative sampling, we hypothesize naive contrastive approaches could still learn spurious correlations (e.g., pulling apart samples different in spurious attribute and class by relying on spurious differences), resulting in lower worst-group accuracy, higher alignment loss, and sample representations with higher mutual information with spurious attributes than default CNC. In Table 3, we find these sampling ablations indeed result in lower worst-group accuracy and less desirable representation metrics.

6. Related work

There is a growing literature on how to improve robustness to spurious correlations, which is a key concern in many settings due to dataset bias against smaller groups. If group labels are known, prior works often design a method to balance groups of different sizes, whether via class balancing (He & Garcia, 2009; Cui et al., 2019), importance weighting (Shimodaira, 2000; Byrd & Lipton, 2019), or robust optimization (Sagawa et al., 2019).

Our work is more related to methods that do not require group labels during training. Such methods commonly first train an initial ERM model, and use this model to train a second robust model. GEORGE (Sohoni et al., 2020) runs GDRO with groups formed by clustering ERM representations. LfF (Nam et al., 2020) and JTT (Liu et al., 2021) train a robust model by upweighting or upsampling the misclassified points of an ERM model. EIIL (Creager et al., 2021) and PGI (Ahmed et al., 2021) infer groups that maximally violate the invariant risk minimization (IRM) objective (Arjovsky & Bottou, 2017) for the ERM model. With these groups EIIL trains a robust model with GDRO, while PGI minimizes the KL divergence of softmaxed logits for same-class samples across groups. CIM (Taghanaki et al., 2021) instead trains a transformation network to remove potentially spurious attributes from image input features. While these approaches can also encourage alignment, our approach more directly acts on a model's representations via contrastive learning. CNC thus leads to better alignment empirically as measured by our representation metrics.

Our proposed algorithm draws inspiration from the literature on self-supervised contrastive learning. which works by predicting whether two inputs are "similar" or "dissimilar" (Le-Khac et al., 2020). This involves specifying batches of *anchor* and *positive* datapoints similar to each other (as different "views" of the same source or input), and *negatives* depicting dissimilar points. In contrastive learning, "negatives" are often sampled uniformly (Bachman et al., 2019), while "positives" are different views of the same object, e.g., via data augmentation (Chen et al., 2020). In supervised contrastive learning, negatives are different-class points and

positives are same-class points (Khosla et al., 2020). Our approach treats same-class points with different ERM predictions as positives, and different-class points with the same ERM prediction as negatives. This naturally provides a form of *hard negative mining*, a nontrivial component of recent contrastive learning shown to improve performance (Robinson et al., 2021; Wu et al., 2021; Chuang et al., 2020). Our approach is also partly inspired by Wang & Isola (2020), who show that minimizing the contrastive loss improves representation alignment between distinct "views." For an expanded discussion of related works, we refer the reader to Appendix D.

7. Conclusion

We present CNC, a two-stage contrastive learning approach to learn representations robust to spurious correlations. We empirically observe and theoretically analyze the connection between alignment and worst-group versus average error. We use this connection to motivate CNC. We find CNC improves the robustness of learned representations by making them more class-dependent and less spurious-attribute-dependent, and achieves SoTA or near-SoTA worst-group accuracy across several benchmarks.

Reproducibility. We make our code publicly available at https://github.com/HazyResearch/correct-n-contrast.

Acknowledgments

We thank Jared Dunnmon, Karan Goel, Khaled Saab, Sabri Eyuboglu, Megan Leszczynski, Laurel Orr, and Sarah Hooper for helpful discussions and feedback.

We gratefully acknowledge the support of NIH under No. U54EB020405 (Mobilize), NSF under Nos. CCF1763315 (Beyond Sparsity), CCF1563078 (Volume to Velocity), and 1937301 (RTML); ARL under No. W911NF-21-2-0251 (Interactive Human-AI Teaming); ONR under No. N000141712266 (Unifying Weak Supervision); ONR N00014-20-1-2480: Understanding and Applying Non-Euclidean Geometry in Machine Learning; N000142012275 (NEPTUNE); Apple, NXP, Xilinx, LETI-CEA, Intel, IBM, Microsoft, NEC, Toshiba, TSMC, ARM, Hitachi, BASF, Accenture, Ericsson, Qualcomm, Analog Devices, Google Cloud, Salesforce, Total, the HAI-GCP Cloud Credits for Research program, the Stanford Data Science Initiative (SDSI), and members of the Stanford DAWN project: Facebook, Google, and VMWare. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views, policies, or endorsements, either expressed or implied, of NIH, ONR, or the U.S. Government.

References

Ahmed, F., Bengio, Y., van Seijen, H., and Courville, A. C. Systematic generalisation with group invariant predictions. In *ICLR*, 2021.

Ahuja, K., Shanmugam, K., Varshney, K., and Dhurandhar, A. Invariant risk minimization games. In *International Conference on Machine Learning*, pp. 145–155. PMLR, 2020.

Alemi, A. A., Fischer, I., Dillon, J. V., and Murphy, K. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016.

Arjovsky, M. and Bottou, L. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

Arjovsky, M., Bottou, L., Gulrajani, I., and Lopez-Paz, D. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Bachman, P., Hjelm, R. D., and Buchwalter, W. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Balashankar, A., Lees, A., Welty, C., and Subramanian, L. What is fair? exploring Pareto-efficiency for fairness constrained classifiers. *arXiv preprint arXiv:1910.14120*, 2019.

Beery, S., Van Horn, G., and Perona, P. Recognition in terra incognita. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 456–473, 2018.

Ben-Tal, A., Den Hertog, D., De Waegenaere, A., Melenberg, B., and Rennen, G. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2):341–357, 2013.

Blodgett, S. L., Green, L., and O'Connor, B. Demographic dialectal variation in social media: A case study of African-American english. *arXiv preprint arXiv:1608.08868*, 2016.

Borkan, D., Dixon, L., Sorensen, J., Thain, N., and Vasserman, L. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion proceedings of the 2019 world wide web conference*, pp. 491–500, 2019.

Buolamwini, J. and Gebru, T. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pp. 77–91. PMLR, 2018.

Byrd, J. and Lipton, Z. What is the effect of importance weighting in deep learning? In *International Conference on Machine Learning*, pp. 872–881. PMLR, 2019.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Chuang, C.-Y., Robinson, J., Lin, Y.-C., Torralba, A., and Jegelka, S. Debiased contrastive learning. In *Advances in Neural Information Processing Systems*, volume abs/2007.00224, 2020.

Combes, R. T. d., Pezeshki, M., Shabanian, S., Courville, A., and Bengio, Y. On the learning dynamics of deep neural networks. *arXiv preprint arXiv:1809.06848*, 2018.

Creager, E., Jacobsen, J.-H., and Zemel, R. Environment inference for invariant learning. In *International Conference on Machine Learning*, pp. 2189–2200. PMLR, 2021.

Cui, Y., Jia, M., Lin, T.-Y., Song, Y., and Belongie, S. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9268–9277, 2019.

Curi, S., Levy, K. Y., Jegelka, S., and Krause, A. Adaptive sampling for stochastic risk-averse learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1036–1047, 2020.

Duchi, J. and Namkoong, H. Variance-based regularization with convex objectives. *The Journal of Machine Learning Research*, 20(1):2450–2504, 2019.

Fang, C., Xu, Y., and Rockmore, D. N. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1657–1664, 2013.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

Goel, K., Gu, A., Li, Y., and Ré, C. Model patching: Closing the subgroup performance gap with data augmentation. In *International Conference on Learning Representations*, 2020.

Gunel, B., Du, J., Conneau, A., and Stoyanov, V. Supervised contrastive learning for pre-trained language model fine-tuning. In *International Conference on Learning Representations*, 2021.

Gutmann, M. and Hyvärinen, A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, 2010.

Hashimoto, T., Srivastava, M., Namkoong, H., and Liang, P. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pp. 1929–1938. PMLR, 2018.

Hassani, K. and Khasahmadi, A. H. Contrastive multi-view representation learning on graphs. In *International Conference on Machine Learning*, pp. 4116–4126. PMLR, 2020.

He, H. and Garcia, E. A. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

Idrissi, B. Y., Arjovsky, M., Pezeshki, M., and Lopez-Paz, D. Simple data balancing achieves competitive worst-group-accuracy. *arXiv preprint arXiv:2110.14503*, 2021.

Kaufman, S., Rosset, S., Perlich, C., and Stitelman, O. Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(4):1–21, 2012.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., and Krishnan, D. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 18661–18673, 2020.

Koh, P. W., Sagawa, S., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., Lee, T., et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pp. 5637–5664. PMLR, 2021.

Krueger, D., Caballero, E., Jacobsen, J.-H., Zhang, A., Binas, J., Zhang, D., Priol, R. L., and Courville, A. Out-of-distribution generalization via risk extrapolation (rex). *arXiv preprint arXiv:2003.00688*, 2020.

Le-Khac, P. H., Healy, G., and Smeaton, A. F. Contrastive representation learning: A framework and review. *IEEE Access*, 8:193907–193934, 2020. doi: 10.1109/ACCESS.2020.3031549.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Levy, D., Carmon, Y., Duchi, J. C., and Sidford, A. Large-scale methods for distributionally robust optimization. In *Advances in Neural Information Processing Systems*, volume 33, pp. 8847–8860, 2020.

Li, H., Pan, S. J., Wang, S., and Kot, A. C. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5400–5409, 2018.

Liu, E. Z., Haghgoo, B., Chen, A. S., Raghunathan, A., Koh, P. W., Sagawa, S., Liang, P., and Finn, C. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, pp. 6781–6792. PMLR, 2021.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 3730–3738, 2015.

Martinez, N., Bertran, M., and Sapiro, G. Minimax pareto fairness: A multi objective perspective. In *International Conference on Machine Learning (ICML)*, 2020.

McInnes, L., Healy, J., and Melville, J. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Mnih, A. and Kavukcuoglu, K. Learning word embeddings efficiently with noise-contrastive estimation. In *NIPS*, 2013.

Nagarajan, V., Andreassen, A., and Neyshabur, B. Understanding the failure modes of out-of-distribution generalization. *arXiv preprint arXiv:2010.15775*, 2020.

Nam, J., Cha, H., Ahn, S., Lee, J., and Shin, J. Learning from failure: De-biasing classifier from biased classifier. In *Advances in Neural Information Processing Systems*, volume 33, pp. 20673–20684, 2020.

Oakden-Rayner, L., Dunnmon, J., Carneiro, G., and Ré, C. Hidden stratification causes clinically meaningful failures in machine learning for medical imaging. In *Proceedings of the ACM conference on health, inference, and learning*, pp. 151–159, 2020.

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Oren, Y., Sagawa, S., Hashimoto, T. B., and Liang, P. Distributionally robust language modeling. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

Parascandolo, G., Neitz, A., Orvieto, A., Gresele, L., and Schölkopf, B. Learning explanations that are hard to vary. *arXiv preprint arXiv:2009.00329*, 2020.

Pezeshki, M., Kaba, S.-O., Bengio, Y., Courville, A., Precup, D., and Lajoie, G. Gradient starvation: A learning proclivity in neural networks. *arXiv preprint arXiv:2011.09468*, 2020.

Ribeiro, M. T., Singh, S., and Guestrin, C. "why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.

Robinson, J., Chuang, C.-Y., Sra, S., and Jegelka, S. Contrastive learning with hard negative samples. In *International Conference on Learning Representations*, 2021.

Saenko, K., Kulis, B., Fritz, M., and Darrell, T. Adapting visual category models to new domains. In *European conference on computer vision*, pp. 213–226. Springer, 2010.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *International Conference on Learning Representations*, 2019.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

Sermanet, P., Lynch, C., Chebotar, Y., Hsu, J., Jang, E., Schaal, S., and Levine, S. Time-contrastive networks: Self-supervised learning from video. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1134–1141, 2018.

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

Sohoni, N., Dunnmon, J., Angus, G., Gu, A., and Ré, C. No subclass left behind: Fine-grained robustness in coarse-grained classification problems. In *Advances in Neural Information Processing Systems*, volume 33, pp. 19339–19352, 2020.

Song, J. and Ermon, S. Multi-label contrastive predictive coding. In *Advances in Neural Information Processing Systems*, volume 33, pp. 8161–8173, 2020.

Taghanaki, S. A., Choi, K., Khasahmadi, A., and Goyal, A. Robust representation learning via perceptual similarity metrics. *arXiv preprint arXiv:2106.06620*, 2021.

Tian, Y., Krishnan, D., and Isola, P. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019.

Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., and Isola, P. What makes for good views for contrastive learning. *arXiv preprint arXiv:2005.10243*, 2020.

von Kügelgen, J., Sharma, Y., Gresele, L., Brendel, W., Schölkopf, B., Besserve, M., and Locatello, F. Self-supervised learning with data augmentations provably isolates content from style. *arXiv preprint arXiv:2106.04619*, 2021.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. The Caltech-UCSD birds-200-2011 dataset. 2011.

Wang, T. and Isola, P. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pp. 9929–9939. PMLR, 2020.

Wang, Z., Simoncelli, E. P., and Bovik, A. C. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pp. 1398–1402. Ieee, 2003.

Wiesemann, W., Kuhn, D., and Sim, M. Distributionally robust convex optimization. *Operations Research*, 62(6): 1358–1376, 2014.

Wu, M., Mosse, M., Zhuang, C., Yamins, D., and Goodman, N. D. Conditional negative sampling for contrastive learning of visual representations. In *International Conference on Learning Representations*, 2021.

Zhang, Z. and Sabuncu, M. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., and Torralba, A. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, 2017.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

A. Contrastive algorithm design details

In this section, we provide further details on the training setup and contrastive batch sampling, algorithmic pseudocode, and additional components related to CNC's implementation.

A.1. Training setup

In Fig. 8, we illustrate the two training stages of Correct-N-Contrast described in Sec. 4. In Stage 1, we first train an ERM model with a cross-entropy loss. For consistency with Stage 2, we depict the output as a composition of the encoder and linear classifier layers. Then in Stage 2, we train a new model with the same architecture using contrastive batches sampled with the Stage 1 ERM model and a supervised contrastive loss (3) (which we compute after the depicted representations are first normalized) to update the encoder layers. Note that unlike prior work in contrastive learning (Chen et al., 2020; Khosla et al., 2020), as we have the class labels of the anchors, positives, and negatives, we also continue forward-passing the unnormalized representations (encoder layer outputs) and compute a cross-entropy loss to update the classifier layers while jointly training the encoder.

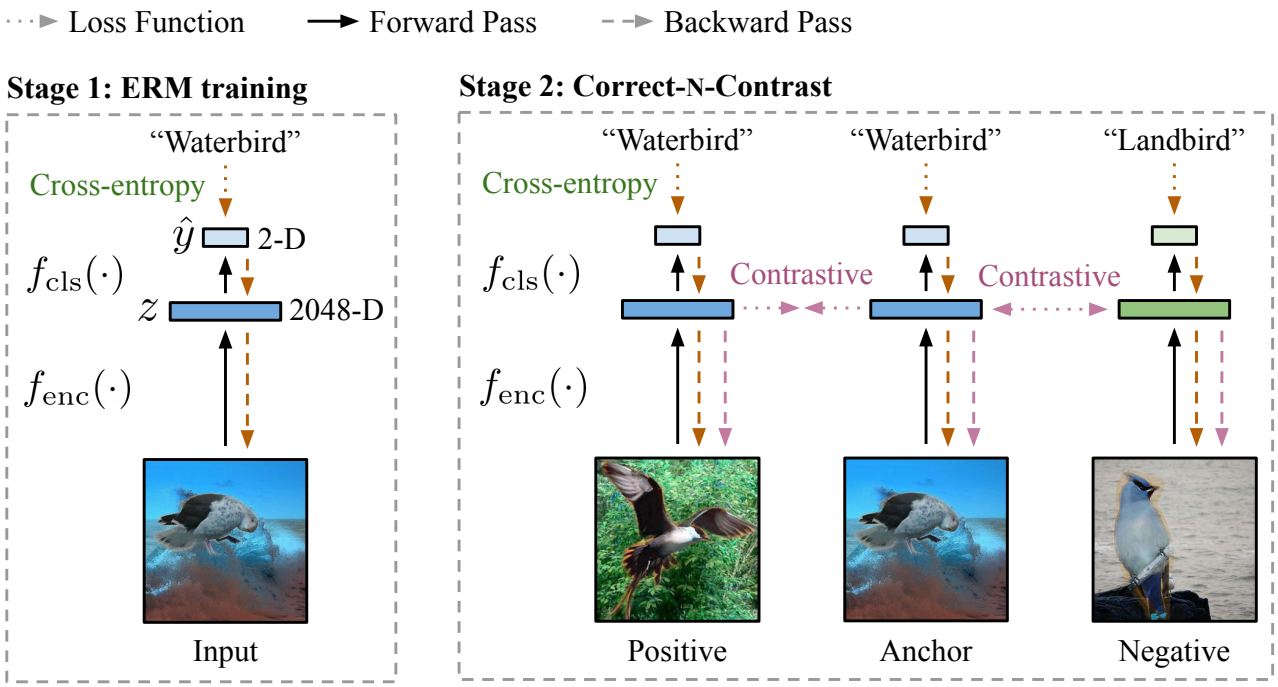


Figure 8. The two stages of Correct-N-Contrast. In Stage 1, we train a model with standard ERM and a cross-entropy loss. Then in Stage 2, we train a new model with the same architecture, but specifically learn spurious-attribute-invariant representations with a contrastive loss (3) and batches of anchors, positives, and negatives sampled with the ERM model's predictions. We also update the full model jointly with a cross-entropy loss on the classifier layer output and the input class labels. Dimensions for ResNet-50 and Waterbirds.

We also note that unlike prior work, we wish to learn invariances between anchors and positives that maximally reduce the presence of features not needed for classification. We thus do not pass the representations through an additional *projection network* (Chen et al., 2020). Instead, we use Eq. 3 to compute the supervised contrastive loss directly on the encoder outputs $z = f_{\text{enc}}(x)$.

A.2. Two-sided contrastive batch implementation

We provide more details on our default contrastive batch sampling approach described in Sec. 4. To recall, for additional contrastive signal per batch, we can double the pairwise comparisons in a training batch by switching the anchor and positive roles. This is similar to the *NT-Xent* loss in prior contrastive learning work (Chen et al., 2020). We switch the role of the anchor and first positive sampled in a contrastive batch, and sample additional positives and negatives using the same guidelines but adjusting for the "new" anchor. We denote this as "two-sided" sampling in contrast with the "one-sided" comparisons we get with just the original anchor, positives, and negatives.

Algorithm 2 Sampling two-sided contrastive batches

Require: Number of positives M and number of negatives N to sample for each batch.

Initialize set of contrastive batches $B = \{\}$
for $x_i \in \{x_i \in X : \hat{y}_i = y_i\}$ **do**
  Sample $M-1$ additional "anchors" to obtain $\{x_i\}_{i=1}^{M}$ from $\{x_i \in X : \hat{y}_i = y_i\}$
  Sample M positives $\{x_m^+\}_{m=1}^{M}$ from $\{x_m^- \in X : \hat{y}_m^- = \hat{y}_i,\ y_m^- \neq y_i\}$
  Sample N negatives $\{x_n^-\}_{n=1}^{N}$ from $\{x_n^- \in X : \hat{y}_n^- = \hat{y}_i,\ y_n^- \neq y_i\}$
  Sample N negatives $\{x_n'^-\}_{n=1}^{N}$ from $\{x_n'^- \in X : \hat{y}_n'^- = \hat{y}_1^+, y_n'^- \neq y_1^+\}$
  Update contrastive batch set: $B \leftarrow B \cup \left(\{x_i\}_{i=1}^{M}, \{x_m^+\}_{m=1}^{M}, \{x_n^-\}_{n=1}^{N}, \{x_n'^-\}_{n=1}^{N}\right)$
end for

Implementing this sampling procedure in practice is simple. First, recall our initial setup with trained ERM model $f_{\hat{\theta}}$, its predictions $\{\hat{y}_i\}_{i=1}^{n}$ on training data $\{(x_i, y_i)\}_{i=1}^{n}$ (where $\hat{y}_i = f_{\hat{\theta}}(x_i)$), and number of positives and negatives to sample M and N. We then sample batches with Algorithm 2.

Because the initial anchors are the datapoints that the ERM model gets correct, under our heuristic we infer $\{x_i\}_{i=1}^{M}$ as samples from the majority group. Similarly the M positives $\{x_m^+\}_{m=1}^{M}$ and N negatives $\{x_n^-\}_{n=1}^{N}$ that it gets incorrect are inferred to belong to minority groups.

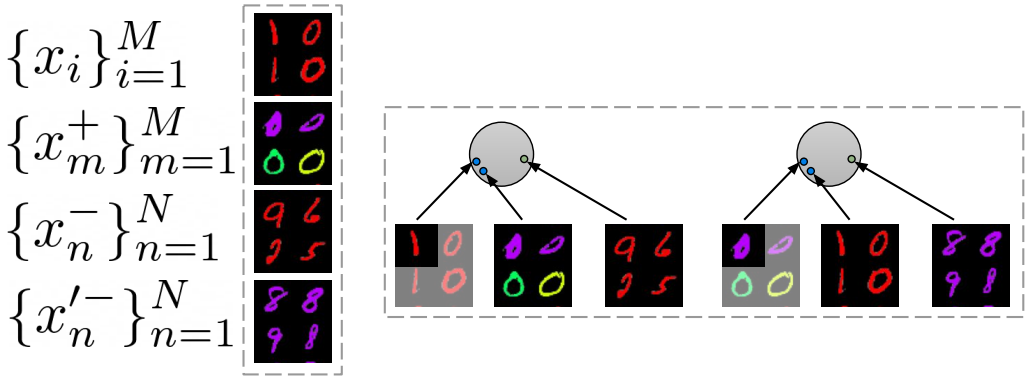
For one batch, we then compute the full contrastive loss with

$$\hat{\mathcal{L}}_{\text{con}}^{\text{sup}}(f_{\text{enc}}) = \hat{\mathcal{L}}_{\text{con}}^{\text{sup}}\left(x_1, \{x_m^+\}_{m=1}^{M}, \{x_n^-\}_{n=1}^{N}; f_{\text{enc}}\right) + \hat{\mathcal{L}}_{\text{con}}^{\text{sup}}\left(x_1^+, \{x_i\}_{i=1}^{M}, \{x_n'^-\}_{n=1}^{N}; f_{\text{enc}}\right) \tag{8}$$

where $\hat{\mathcal{L}}_{\text{con}}^{\text{sup}}\left(x_1, \{x_m^+\}_{m=1}^{M}, \{x_n^-\}_{n=1}^{N}; f_{\text{enc}}\right)$ is given by:

$$-\frac{1}{M}\sum_{m=1}^{M} \log \frac{\exp(z_1^\top z_m^+/\tau)}{\sum_{m=1}^{M}\exp(z_1^\top z_m^+/\tau) + \sum_{n=1}^{N}\exp(z_1^\top z_n^+/\tau)} \tag{9}$$

and again let z be the normalized output $f_{\text{enc}}(x)$ for corresponding x. We compute the cross-entropy component of the full loss for each x in the two-sided batch with its corresponding label y.



(a) Contrastive Batch (b) Two-sided Update

Figure 9. Illustration of two-sided contrastive batch sampling with Colored MNIST as an example. From a single batch (a), we can train a contrastive model with two anchor-positive-negative pairings (b). Aside from increasing the number of "hard negatives" for each anchor-positive pair, this intuitively "pushes" together anchors and positives from two different directions for greater class separation.

A.3. Summary of CNC design choices and properties

No projection network. As we wish to learn data representations that maximize the alignment between anchor and positive datapoints, we do not compute the contrastive loss with the outputs of an additional nonlinear projection network. This is inspired by the logic justifying a projection head in prior contrastive learning, e.g. SimCLR (Chen et al., 2020), where the head is included because the contrastive loss trains representations to be "invariant to data transformation" and may encourage removing information "such as the color or orientation of objects". In our case, we view inferred datapoints with the same class but different spurious attributes as "transformations" of each other, and we hypothesize that removing these differences can help us improve worst-group performance.

Two-sided contrastive sampling. To incorporate additional comparisons between datapoints that only differ in spurious attribute during training, we employ "two-sided" contrastive batch sampling. This lets us equally incorporate instances where the second contrastive model in CNC treats datapoints that the initial ERM model got incorrect and correct as anchors.

Additional intrinsic hard positive/negative mining. Because the new model corrects for potentially learned spurious correlations by only comparing and contrasting datapoints that differ in class label or spurious attribute, but not both (as dictated by the initial ERM model's outputs), the contrastive batches naturally carry "hard" positives and negatives. Thus, our approach provides a natural form of hard negative mining (in addition to the intrinsic hard positive / negative mining at the gradient level with InfoNCE-style contrastive losses (Chen et al., 2020; Khosla et al., 2020)) while avoiding class collisions, two nontrivial challenges in standard self-supervised contrastive learning (Robinson et al., 2021; Wu et al., 2021; Chuang et al., 2020).

Joint training of encoder and classifier layers. CNC can train any standard classification model architecture; for any given neural network we just apply different optimization objectives to the encoder and classifier layers. We train both the encoder and classifier layers with a cross-entropy loss, and jointly train the encoder layer with a supervised contrastive loss. For the encoder layers, we balance the two objectives with a hyperparameter λ (c.f. Eq. 7).

B. Omitted Proofs from Section 3.2

In this section, we prove that within any class, the gap between the worst-group error and the average error can be upper bounded by the alignment loss times the Lipschitz constant, plus another concentration error term.

Proof of Theorem 3.1. Consider two arbitrary groups, denoted by $g_1 = (y, a_1)$ and $g_2 = (y, a_2)$, whose class labels are both $y \in \mathcal{Y}$, whose spurious attributes are $a_1 \in \mathcal{A}$ and $a_2 \in \mathcal{A}$ such that $a_1 \neq a_2$. Let G_1 and G_2 be the subset of training data that belong to groups g_1 and g_2, respectively. We note that both G_1 and G_2 are non-empty since we have assumed that (in Section 2) there is at least one sample from each group in the training data set. Let $n_{g_1} = |G_1|$ and $n_{g_2} = |G_2|$ be the size of these two groups, respectively. Recall that f_{enc} denotes the mapping of the encoder layers of the full neural network model f_θ. Since the classification layer f_{cls} is a linear layer, we have used W to denote the weight matrix of this layer. Our definition of the cross-group alignment loss in equation (5), denoted as $\hat{\mathcal{L}}_{\text{align}}(f_\theta; y)$, implies that for g_1 and g_2,

$$\frac{1}{n_{g_1}} \frac{1}{n_{g_2}} \sum_{(x,y,a_1)\in G_1} \sum_{(x',y,a_2)\in G_2} \|f_{\text{enc}}(x) - f_{\text{enc}}(x')\|_2 \leq \hat{\mathcal{L}}_{\text{align}}(f_\theta; y). \tag{10}$$

Next, let $\mathbb{E}_{(x,y,a_1)\sim\mathcal{P}_{g_1}}[\mathcal{L}_{\text{avg}}(W f_{\text{enc}}(x), y)]$ be the average loss conditioning on a data point being sampled from group g_1 (and similarly for group g_2). Let $\Delta(g_1, g_2)$ be the difference between the population average losses:

$$\Delta(g_1, g_2) = \left| \mathop{\mathbb{E}}_{(x,y,a_1)\sim\mathcal{P}_{g_1}} [\mathcal{L}_{\text{avg}}(W f_{\text{enc}}(x), y] - \mathop{\mathbb{E}}_{(x,y,a_2)\sim\mathcal{P}_{g_2}} [\mathcal{L}_{\text{avg}}(W f_{\text{enc}}(x), y)] \right|.$$

Recall that $\mathcal{G}_y \subseteq \mathcal{G}$ is the set of groups that have class label y. Since the loss $\ell(\cdot)$ is bounded above by some fixed constant C_2 according to our assumption, and is at least zero, by the Hoeffding's inequality, the following result holds with probability at least $1 - \delta$, for all $|\mathcal{G}_y|$ groups $g \in \mathcal{G}_y$,

$$\left| \mathop{\mathbb{E}}_{(x,y,a)\sim\mathcal{P}_g} [\mathcal{L}_{\text{avg}}(W f_{\text{enc}}(x), y)] - \frac{1}{n_g} \sum_{(x,y)\in(X,Y)} \ell(W f_{\text{enc}}(x), y) \right| \leq C_2 \sqrt{\frac{2 \log\left(|\mathcal{G}_y| / \delta\right)}{n_g}}. \tag{11}$$

Thus, with probability at least $1-\delta$, the following holds for any g_1 and g_2 in class y (but having different spurious attributes)

$$\Delta(g_1, g_2) \leq \left| \frac{1}{n_{g_1}} \sum_{(x,y,a_1)\in G_1} \mathcal{L}_{\text{avg}}(W f_{\text{enc}}(x), y) - \frac{1}{n_{g_2}} \sum_{(x',y,a_2)\in G_2} \mathcal{L}_{\text{avg}}(W f_{\text{enc}}(x'), y) \right| + C_2 \left(\sqrt{\frac{2\log(|\mathcal{G}_y|/\delta)}{n_{g_1}}} + \sqrt{\frac{2\log(|\mathcal{G}_y|/\delta)}{n_{g_2}}} \right). \tag{12}$$

Next, we focus on the RHS of equation (12). First, equation (12) is also equal to the following:

$$\left| \frac{1}{n_{g_1}} \frac{1}{n_{g_2}} \sum_{(x,y,a_1)\in G_1} \sum_{(x',y,a_2)\in G_2} \ell(W f_{\text{enc}}(x), y)) - \frac{1}{n_{g_1}} \frac{1}{n_{g_2}} \sum_{(x,y,a_1)\in G_1} \sum_{(x',y,a_2)\in G_2} \ell(W f_{\text{enc}}(x'), y)) \right|.$$

Since we have also assumed that the loss function $\ell(x, y)$ is C_1-Lipschitz in x[3], the above is at most:

$$\begin{aligned}
&\left| \frac{1}{n_{g_1} n_{g_2}} \sum_{(x,y,a_1)\in G_1} \sum_{(x',y,a_2)\in G_2} |\ell(W f_{\text{enc}}(x), y) - \ell(W f_{\text{enc}}(x'), y)| \right| \\
\leq &\frac{1}{n_{g_1} n_{g_2}} \sum_{(x,y,a_1)\in G_1} \sum_{(x',y,a_2)\in G_2} C_1 \cdot \|W f_{\text{enc}}(x) - W f_{\text{enc}}(x')\|_2 && \text{(since } y \text{ is the same for } x, x') \\
\leq &\frac{B}{n_{g_1} n_{g_2}} \sum_{(x,y,a_1)\in G_1} \sum_{(x',y,a_2)\in G_2} C_1 \cdot \|f_{\text{enc}}(x) - f_{\text{enc}}(x')\|_2 && \text{(because } \|W\|_2 \leq B \text{ as assumed)} \\
\leq &B \cdot C_1 \cdot \hat{\mathcal{L}}_{\text{align}}(f_\theta; y). && \text{(because of equation (10))}
\end{aligned}$$

Thus, we have shown that for any g_1 and g_2 within class y,

$$\begin{aligned}
\Delta(g_1, g_2) &\leq B \cdot \hat{\mathcal{L}}_{\text{align}}(f_\theta; y) + \left(\sqrt{\frac{2\log(|\mathcal{G}_y|/\delta)}{n_{g_1}}} + \sqrt{\frac{2\log(|\mathcal{G}_y|/\delta)}{n_{g_2}}} \right) \\
&\leq B \cdot C_1 \cdot \hat{\mathcal{L}}_{\text{align}}(f_\theta; y) + \max_{g\in\mathcal{G}_y} C_2 \cdot \sqrt{\frac{8\log(|\mathcal{G}_y|/\delta)}{n_g}}.
\end{aligned} \tag{13}$$

Finally, we use the above result to bound the gap between the worst-group loss and the average loss. For every group $g \in \mathcal{G}$, let p_g denote the prior probability of observing a sample from $\mathcal{P}$ in this group. Let $q_y = \sum_{g'\in\mathcal{G}_y} p_{g'}$. Let $h(g)$ be a short hand notation for

$$h(g) = \mathop{\mathbb{E}}_{(x,y,a)\sim\mathcal{P}_g} \left[\mathcal{L}_{\text{avg}}(W f_{\text{enc}}(x), y) \right].$$

The average loss among the groups with class label y is $\mathcal{L}_{\text{avg}}(f_\theta; y) = \sum_{g\in\mathcal{G}_y} \frac{p_g}{q_y} h(g)$. The worst-group loss among the groups with class label y is $\mathcal{L}_{\text{wg}}(f_\theta; y) = \max_{g\in\mathcal{G}_y} h(g)$. Let $g^\star$ be a group that incurs the highest loss among groups in $\mathcal{G}_y$. We have $\mathcal{L}_{\text{wg}}(f_\theta; y) - \mathcal{L}_{\text{avg}}(f_\theta; y)$ is equal to

$$h(g^\star) - \sum_{g\in\mathcal{G}_y} \frac{p_g}{q_y} h(g) = \sum_{g\in\mathcal{G}_y} \frac{p_g}{q_y} (h(g^\star) - h(g)) \tag{14}$$

$$\leq \sum_{g\in\mathcal{G}_y} \frac{p_g}{q_y} \Delta(g^\star, g) \tag{15}$$

$$\leq B \cdot C_1 \cdot \hat{\mathcal{L}}_{\text{align}}(f_\theta; y) + \max_{g\in\mathcal{G}_y} C_2 \cdot \sqrt{\frac{8\log(|\mathcal{G}|/\delta)}{n_g}}. \tag{16}$$

The last step uses equation (13) on $\Delta(g^\star, g)$ and the fact that $q_y = \sum_{g'\in\mathcal{G}_y} p_{g'}$. Thus, we have shown that the gap between the worst-group loss and the average loss among the groups with the same class label is bounded by the above equation. The proof is now complete. □

[3] In other words, we assume that $|\ell(z, y) - \ell(z', y)| \leq C_1 \cdot \|z - z'\|_2$, for any z, z' and y.

The astute reader will note that Theorem 3.1 focuses on comparing groups within the same class y, for any $y \in \mathcal{Y}$. A natural follow-up question is what happens when comparing across groups with different labels. Let $\mathcal{L}_{\text{wg}}(f_\theta) = \max_{y\in\mathcal{Y}} \mathcal{L}_{\text{wg}}(f_\theta; y)$ be the worst-group loss across all the labels. Recall that $\mathcal{L}_{\text{avg}}(f_\theta)$ is the average loss for the entire population of data. We generalize Theorem 3.1 to this setting in the following result.

Corollary B.1 (Extension of Theorem 3.1 to compare across different classes)**.** *In the setting of Theorem 3.1, let $q_y = \sum_{g\in\mathcal{G}_y} p_g$ be the prior probability of observing a sample drawn from $\mathcal{P}$ with label y, for any $y \in \mathcal{Y}$. We have that with probability at least $1-\delta$, the following holds:*

$$\mathcal{L}_{wg}(f_\theta) \leq \left(\min_{y\in\mathcal{Y}} q_y\right)^{-1} \mathcal{L}_{avg}(f_\theta) + B\cdot C_1 \cdot \max_{y\in\mathcal{Y}} \hat{\mathcal{L}}_{align}(f_\theta; y) + \max_{g\in\mathcal{G}} C_2 \cdot \sqrt{\frac{8\log(|\mathcal{G}|/\delta)}{n_g}}. \tag{17}$$

Proof. We generalize the argument in the previous result to compare across different labels. The worst-group loss across different labels is

$$\begin{aligned}
&\max_{y\in\mathcal{Y}} \max_{g\in\mathcal{G}_y} h(g) \\
\leq &\max_{y\in\mathcal{Y}} \left(\sum_{g\in\mathcal{G}_y} \frac{p_g}{q_y} h(g) + B\cdot C_1 \hat{\mathcal{L}}_{\text{align}}(f_\theta; y) + \max_{g\in\mathcal{G}_y} C_2 \sqrt{\frac{8\log(|\mathcal{G}_y|/\delta)}{n_g}} \right) && \text{(because of equation (16))} \\
\leq &\frac{1}{\min_{y\in\mathcal{Y}} q_y} \sum_{g\in\mathcal{G}} p_g h(g) + B\cdot C_1 \max_{y\in\mathcal{Y}} \hat{\mathcal{L}}_{\text{align}}(f_\theta; y) + \max_{g\in\mathcal{G}} C_2 \sqrt{\frac{8\log(|\mathcal{G}|/\delta)}{n_g}}.
\end{aligned}$$

Since $\sum_{g\in\mathcal{G}} p_g h(g) = \mathcal{L}_{\text{avg}}(f_\theta)$, we thus conclude that

$$\mathcal{L}_{\text{wg}}(f_\theta) \leq \left(\min_{y\in\mathcal{Y}} q_y\right)^{-1} \mathcal{L}_{\text{avg}}(f_\theta) + B\cdot C_1 \max_{y\in\mathcal{Y}} \hat{\mathcal{L}}_{\text{align}}(f_\theta; y) + \max_{g\in\mathcal{G}} C_2 \sqrt{\frac{8\log(|\mathcal{G}|/\delta)}{n_g}}.$$

The proof is now complete. □

An example showing that Corollary B.1 is tight. We describe a simple example in which the factor $\left(\min_{y\in\mathcal{Y}} q_y\right)^{-1}$ in equation (17) is tight (asymptotically). Suppose there are k perfectly balanced classes so that $q_y = 1/k$, for every $y \in \mathcal{Y}$. There is one data point from each class, with loss equal to 0 for all except one of them. The worst-group loss is 1 whereas the average loss is $1/k$. Thus, there is a factor of k between the worst-group loss and the average loss. For equation (17), the factor

$$\left(\min_{y\in\mathcal{Y}} q_y\right)^{-1} = k,$$

since $q_y = 1/k$ for every $y \in \mathcal{Y}$ in this example. Thus, this factor matches the (multiplicative) factor between the worst-group loss and the average loss in this example.

C. Additional empirical comparisons and ablations

In this section, we include further experiments comparing CNC against additional related methods and ablations to study the importance of CNC's presented design choices. We first consider an alternative representation alignment procedure by minimizing the presented alignment loss directly as opposed to using contrastive learning in Appendix C.1. We next report additional mutual information metrics from our experiments in Section 5.2, where we study the relation between these representation metrics and worst-group accuracy on datasets with increasingly strong spurious correlations. We then summarize and empirically ablate various other design choices for CNC in Appendix C.3. In Appendix C.6, we finally compare CNC against related work in representation learning for unsupervised domain adapation—a problem setting that similarly involves overcoming group-specific dependencies—once properly adapted for our spurious correlations setting, .

C.1. Comparison to minimizing the alignment loss directly

In Sec. 5.1 and Sec. 5.2, we empirically showed that CNC's contrastive loss and hard positive and negative sampling lead to improved worst-group accuracy and greater representation alignment. While the analysis in Wang & Isola (2020) also

discusses how contrastive learning supports alignment between anchor and positives, we also investigate how CNC performs if instead of the contrastive loss, we train the Stage 2 model to minimize $\mathcal{L}_{\text{align}}$ directly as a training objective. With this objective, we aim to minimize the Euclidean distance between samples in different inferred groups but the same class. While CNC is motivated by improving alignment, we hypothesize that one advantage of the contrastive loss lies in not only aligning anchors and positives, but also *pulling apart* hard negatives, improving class-separability. We keep all other components consistent, and apply $\mathcal{L}_{\text{align}}$ to the anchor and positive samples in each contrastive batch. We report results on Waterbirds and CelebA in Table 5.

Table 5. Accuracy (%) and representation metrics ($\times 10$) comparing CNC as proposed vs. $\mathcal{L}_{\text{align}}$ as the training objective. While the latter results in lower alignment, it does not encourage separating hard negatives from anchors. This results in representations with lower mutual information with class labels and higher mutual information with spurious attributes, and lower worst-group and average accuracies.

	Waterbirds					CelebA				
Method	Avg. Acc.	WG Acc.	$\mathcal{L}_{\text{align}}$ [↓]	$\hat{I}(Y;Z)$ [↑]	$\hat{I}(A;Z)$ [↓]	Avg. Acc.	WG Acc.	$\mathcal{L}_{\text{align}}$ [↓]	$\hat{I}(Y;Z)$ [↑]	$\hat{I}(A;Z)$ [↓]
CNC ($\mathcal{L}_{\text{align}}$)	82.3 (0.1)	88.9 (0.0)	**2.16 (0.01)**	2.52 (0.06)	3.27 (0.02)	82.3 (1.6)	85.9 (0.8)	**2.59 (0.20)**	2.00 (0.03)	1.59 (0.30)
CNC	**90.9 (0.1)**	**88.5 (0.3)**	6.02 (0.35)	3.56 (0.04)	**0.02 (0.01)**	**89.9 (0.5)**	**88.8 (0.9)**	3.00 (0.12)	**2.44 (0.06)**	**1.45 (0.18)**

We find that CNC with the default contrastive loss outperforms CNC with the alignment loss in both worst-group and average accuracy, and that this indeed corresponds with representations that exhibit lower mutual information with class labels. We reason that the additional pushing apart of hard negative samples (with different class labels but similar spurious features) provides additional signal for improving separation between the different classes. The robust model thus only learns to rely on class-specific features for discriminating between datapoints. On the other hand, the $\mathcal{L}_{\text{alignment}}$ objective does not incorporate these hard negatives.

C.2. Extended analysis of representation metrics over increasing spurious correlations

In Section 5.2, we found that CNC's improved worst-group accuracy on popular spurious correlations benchmarks coincided with representations with lower alignment loss, higher mutual information with ground-truth classes, and lower mutual information with spurious attributes. To study how this relation between representation metrics and worst-group accuracy scaled with the strength of the spurious correlation, we also computed representation metrics over increasingly spurious CMNIST* datasets for CNC, ERM, and JTT. Below in Fig. 10 we reproduce Fig. 7 with added mutual information metrics.

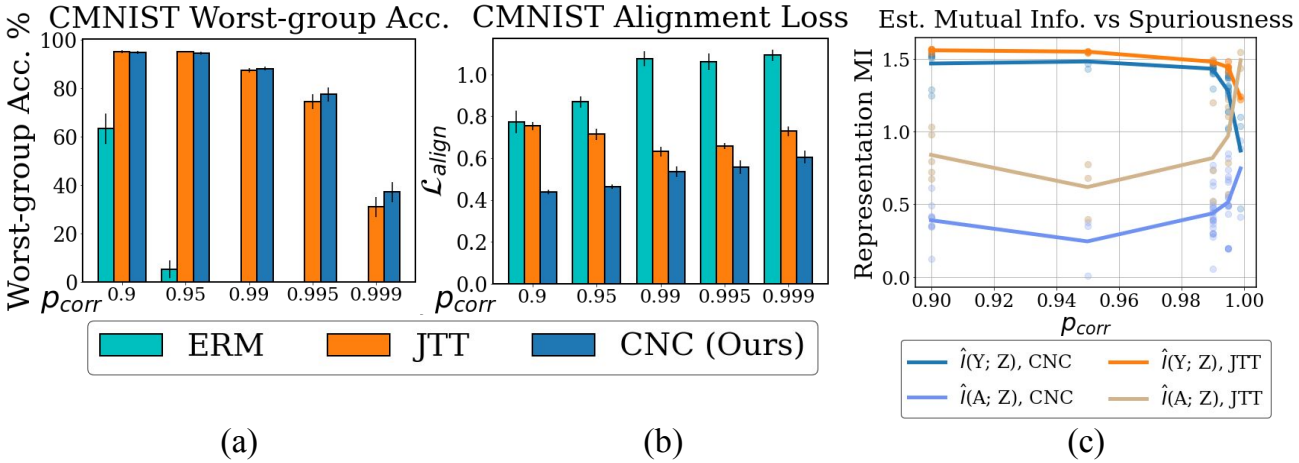


Figure 10. Alignment loss and mutual information metrics with worst-group accuracy on increasingly spurious CMNIST*. CNC's highest worst-group accuracy in spuriously correlated datasets (a) coincides with learning representations with better alignment (b), and a more desirable ratio of mutual information dependence on the ground-truth class labels vs the spurious attribute (c).

Fig. 10(c) shows that CNC's learned representations maintain a more favorable balance of mutual information between the class label and spurious attribute than JTT. While JTT models exhibit slightly higher estimated $\hat{I}(Y;Z)$ than CNC models, CNC models exhibit much lower dependence on the spurious attribute. In the regime where 99.9% of digits are spuriously correlated with a specific color, CNC uniquely learns representations with higher mutual information on class labels than spurious attributes.

C.3. Additional design choice ablations

To validate the additional algorithmic components of CNC, we report how CNC performs on the Waterbirds dataset when modifying the individual design components. We use the same hyperparameters as in the main results, and report accuracies as the average over three training runs for the following ablations. Table 6 summarizes that across these design ablations, default CNC as presented consistently outperforms these alternative implementations.

Table 6. Ablation over CNC algorithmic components on Waterbirds. Default choices achieve highest worst-group and average accuracy.

Method	CNC (Default)	Projection Head	One-sided Contrasting	Train + Finetune
WG Acc. (%)	**88.5 (0.3)**	82.4 (1.8)	85.2 (3.6)	84.0 (1.7)
Avg. Acc. (%)	**90.9 (0.1)**	88.7 (0.6)	90.1 (1.6)	87.7 (1.1)

No projection head. We incorporate a nonlinear projection head as is typical in prior contrastive learning works (Chen et al., 2020), that maps the encoder output to lower-dimensional representations (from 2048 to 128 in our case). We then update the encoder layers and the projection head jointly by computing the contrastive loss on the projection head's output, still passing the encoder layer's direct outputs to the classifier to compute the cross-entropy loss. We note that using the projection head decreases worst-group accuracy substantially. We reason that as previously discussed, while using the projection head in prior work can allow the model to retain more information in its actual hidden layers (Chen et al., 2020), in our case to remove dependencies on spurious attributes we actually want to encourage learning invariant representations when we model the differences between anchor and positive datapoints as due to spurious attributes.

Two-sided contrastive batches. Instead of "two-sided" contrasting where we allow both sampled anchors and positives to take on the anchor role, for each batch we only compute contrastive updates by comparing original positives and negatives with the original anchor. When keeping everything else the same, we find that just doing these one-sided comparisons also leads to a drop in performance for worst-group accuracy. This suggests that the increased number of comparisons where we swap the roles of anchors and positives introduces greater contrastive learning signal.

Joint training of encoder and classifier layers. Instead of training the full model jointly, we first only train the encoder layers with the contrastive loss in CNC, before freezing these layers and finetuning the classifier layers with the cross-entropy loss. With this implementation, we also obtain noticeable drop in performance. While we leave further analysis for the joint cross-entropy and contrastive optimization for future work, one conjecture is that the cross-entropy loss may aid in learning separable representations while also training the full model to keep the average error small.

This also follows prior work, where updating the entire model and finetuning all model parameters instead of freezing the encoder layers leads to higher accuracy (Chen et al., 2020). However, we found that with an initial encoder-only training stage, if we did not freeze the trained layers the fine-tuning on a dataset with spurious correlations would "revert" the contrastive training, resulting in a large gap between worst-group and average error similar to ERM.

Contrastive loss balancing hyperparameter. We also ablate the balancing hyperparameter λ of CNC on CMNIST*. In Table 7 we find that CNC consistently achieves high worst-group accuracy across a wide range of $\lambda \in [0.4, 0.9]$. For reference, the next best methods GEORGE and JTT obtain 76.4% and 74.5% worst-group accuracy.

Table 7. Ablation over λ to balance cross-entropy and contrastive loss on CMNIST*. CNC obtains high performance across a range of λ.

CNC λ	0.2	0.4	0.6	0.8	0.9
Robust Acc.	70.4 (2.9)	74.2 (2.6)	75.3 (1.7)	77.4 (2.5)	75.8 (1.2)
Average Acc.	89.0 (0.1)	88.0 (0.7)	88.3 (0.6)	89.9 (0.4)	88.4 (0.1)

C.4. Comparison to recent class-balancing baselines for improving worst-group accuracy

In Table 8, we compare CNC and ERM with reported worst-group accuracies with recent robustness baselines from Idrissi et al. (2021) that do not require training group labels. These involve subsampling large classes (SUBY) and reweighting classes to balance classes per minibatch in expectation (RWY). These baselines improve worst-group accuracy over ERM, but CNC achieves significantly higher worst-group accuracy across reported dataset results.

C.5. Comparison to model selection by average accuracy

As in prior work (Sagawa et al., 2019), for our main results we select models with early stopping based on best worst-group validation accuracy. This involves using validation set group labels, which may not always be available. In Table 9, we

Table 8. Worst-group accuracy (%) of CNC (mean and standard deviation over three seeds) and reported baselines from Idrissi et al. (2021) (same metrics, over five seeds), which subsample (SUBY) or reweight (RWY) samples to balance classes during training.

	Waterbirds	CelebA	CivilComments-WILDS
ERM	62.6 (0.3)	47.7 (2.1)	58.6 (1.7)
SUBY	82.4 (1.7)	79.9 (3.3)	51.2 (3.0)
RWY	86.1 (0.7)	82.9 (2.2)	67.5 (0.6)
CNC	**88.5 (0.3)**	**88.8 (0.9)**	**68.9 (2.1)**

report how CNC performs when we do model selection with early stopping based on best *average* validation accuracy. We compare report the differences between model selection for both worst-group and average accuracy. Without any group labels, CNC's worst-group accuracy drops, but CNC is still obtains much higher worst-group accuracy over ERM.

Table 9. Worst-group accuracy (%) of CNC (mean and standard deviation over three seeds) for CNC models selected based on best validation worst-group accuracy (as in prior work) compared to CNC models based on best validation *average* accuracy.

	Waterbirds		CelebA	
	Worst-group	Average	Worst-group	Average
Best Val Worst-group Acc.	88.5 (0.3)	90.9 (0.1)	88.8 (0.9)	89.9 (0.5)
Best Val Average Acc.	85.0 (1.8)	94.7 (0.2)	87.6 (0.6)	90.8 (0.4)
Difference	−3.5	+3.8	−1.2	+0.9

C.6. Comparison to representation learning methods for domain generalization

While our main results in Table 1 compare against methods designed to tackle the spurious correlations setting presented in Section 5.1, we also note that CNC bears some similarity to methods proposed for unsupervised domain adaptation (UDA). At a high level, a popular approach is to learn similar representations for datapoints with the same class but sampled from different domains, e.g., via adversarial training to prevent another model from classifying representations' source domains correctly (Ganin et al., 2016), or minimizing representation differences via metrics such as *maximum mean discrepancy* (MMD) (Li et al., 2018), to generalize to a desired target domain. While UDA carries distinct problem settings and assumptions from our spurious correlations setting (c.f. Appendix D.4), we aim to understand if UDA methods can that also try to optimize a model's representations can train models robust to spurious correlations, and compare their performance with CNC. We first explain our protocol for fair evaluation, and then discuss results reported in Table 10.

We carry out our evaluation with domain-adversarial neural networks (DANN) (Ganin et al., 2016), a seminal DG method that aims to learn aligned representations across two domains. To do so, DANN jointly trains a model to classify samples from a "source" domain while preventing a separate "domain classifier" module from correctly classifying the domain for datapoints sampled from both domains. For fair comparison, we use the same ResNet-50 backbone as in CNC, and make several adjustments to the typical DANN and UDA procedure:

1. While UDA assumes that the data is organized into "source" and "target" domains, we do not have domain labels. We thus infer domains using the predictions of an initial ERM model as in CNC.
2. The notion of a domain may also be ambiguous with respect to the groups defined in Section 2. For example, domains may be defined by spurious attributes (e.g., for the Waterbirds dataset, we may consider the "water background" domain and the "land background" domain). Domains may alternatively be defined by whether samples carry dominant spurious correlations or not (e.g., the "majority group" domain and the "minority group" domain). We train and evaluate separate DANN models for both interpretations. We infer the former by the predicted class of the initial ERM model. We infer the latter by whether the initial ERM model is correct or not.
3. Finally, UDA aims to train with a class-labeled "source" domain and an unlabeled "target" domain such that a model performs well on unseen samples from the specified "target" domain (Ganin et al., 2016). However, our benchmarks have class labels for *all* training points, and do not have a notion of "source" and "target" domains (we aim to obtain high worst-group accuracy, which could fall under any domain). We thus assume access to labels for all domains. During training, the goal for our DANN models is to correctly classify samples from both domains, while learning representations such that a jointly trained domain classifier module cannot determine the samples' domains from their representations alone. At test-time, we evaluate the DANN model on the entire test set for each benchmark, and report the worst-group and average accuracies.

In Table 10, we report the worst-group and average accuracies of DANN on the Waterbirds and CelebA datasets across three seeds along with the CNC results. Our results suggest that the domain alignment in DANN is not sufficient to improve worst-group accuracy. We hypothesize this is due to adversarial training with the domain classifier aligning representations without regard to different classes within each domain. Due to the propensity of samples exhibiting spurious correlations, DANN models may thus still learn to rely on these correlations.

Table 10. CNC achieves higher worst-group and average accuracies on spuriously correlated benchmarks than DANN, a prior representation alignment method designed for domain adaptation

Method	Waterbirds		CelebA	
Accuracy (%)	Worst-group	Average	Worst-group	Average
DANN (domains by spurious attribute)	37.4 (3.8)	87.6 (2.2)	28.1 (3.1)	94.6 (0.3)
DANN (domains by majority vs minority group)	67.3 (0.8)	83.6 (0.2)	47.2 (3.1)	88.7 (1.8)
CNC	**88.5 (0.3)**	**90.9 (0.1)**	**88.8 (0.9)**	**89.9 (0.5)**

D. Further related work discussion

We provide additional discussion of related work and connections to our work below.

D.1. Improving robustness to spurious correlations

Our core objective is to improve model robustness to group or subpopulation distribution shifts that arise from the presence of spurious correlations, specifically for classification tasks. Because these learnable correlations hold for some but not all samples in a dataset, standard training with ERM may result in highly variable performance: a model that classifies datapoints based on spurious correlations does well for some subsets or "groups" of the data but not others. To improve model robustness and avoid learning spurious correlations, prior work introduces the goal to maximize worst-group accuracy (Sagawa et al., 2019). Related works broadly fall under two categories:

Improving robustness with group information. If information such as spurious attribute labels is provided, one can divide the data into explicit groups as defined in Sec. 2, and then train to directly minimize the worst group-level error among these groups. This is done in group DRO (GDRO) (Sagawa et al., 2019), where the authors propose an online training algorithm that focuses training updates over datapoints from higher-loss groups. Goel et al. (2020) also adopt this approach with their method CycleGAN Augmented Model Patching (CAMEL). However, similar to our motivation, they argue that a stronger modeling goal should be placed on preventing a model from learning group-specific features. Their approach involves first training a CycleCAN (Zhu et al., 2017) to learn the data transformations from datapoints in one group to another that share the same class label. They then apply these transformations as data augmentations to different samples, intuitively generating new versions of the original samples that take on group-specific features. Finally they train a new model with a consistency regularization objective to learn invariant features between transformed samples and their sources. Unlike their consistency loss, we accomplish a similar objective to learn group-invariant features with contrastive learning. Our first training stage is also less expensive. Instead of training a CycleGAN and then using it to augment datapoints, we train a relatively simple standard ERM classification model, sometimes with only a few number of epochs, and use its predictions to identify pairs of datapoints to serve a similar purpose. Finally, unlike both CAMEL and GDRO, we do not require spurious attribute or group labels for each training datapoints. We can then apply CNC in less restrictive settings where such information is not known.

Related to GDRO are methods that aim to optimize a "Pareto-fair" objective, more general than simply the worst-case group performance. Notable examples are the works of Balashankar et al. (2019) and Martinez et al. (2020). However, these approaches similarly do not directly optimize for good representation alignment, which is a focus of our work.

Improving robustness without training group information. More similar to our approach are methods that do not assume group information at training time, and only require validation set spurious attribute labels for fine-tuning. As validation sets are typically much smaller in size than training sets, an advantage of CNC and comparable methods is that we can improve the accessibility of robust training methods to a wider set of problems. One popular line of work is distributionally robust optimization (DRO), which trains models to minimize the worst loss within a ball centered around the observed distribution (Ben-Tal et al., 2013; Wiesemann et al., 2014; Duchi & Namkoong, 2019; Levy et al., 2020; Curi et al., 2020; Oren et al., 2019). However, prior work has shown that these approaches may be too pessimistic, optimizing not just for worst-group accuracy but worst possible accuracy within the distribution balls (Sagawa et al., 2019), or too undirected, optimizing for

too many subpopulations, e.g. by first upweighting minority points but then upweighting majority points in later stages of training (Liu et al., 2021). Pezeshki et al. (2020) instead suggest that *gradient starvation* (GS), where neural networks only learn to capture statistically dominant features in the data (Combes et al., 2018), is the main culprit behind learning spurious correlations, and introduce a "spectral decoupling" regularizer to alleviate GS. However this does not directly prevent models from learning dependencies on spurious attributes. Similar to CAMEL, (Taghanaki et al., 2021) propose Contrastive Input Morphing (CIM), an image dataset-specific method that aims to learn input feature transformations that remove the effects of spurious or task-irrelevant attributes. They do so without group labels, training a transformation network with a triplet loss to transform input images such that a given transformed image's *structural similarity metric* (based on luminance, contrast, and structure (Wang et al., 2003)) is more similar to a "positive" image from the same class than a "negative" image from a different class. They then train a classifier on top of these representations. Instead of pixel-level similarity metrics, CNC enforces similarity in a neural network's hidden-layer representations, allowing CNC to apply to non-image modalities. Additionally, we sample positives and negatives not just based on class label, but also the learned spurious correlations of an ERM model (via its trained predictions). We hypothesize that our sampling scheme, which intuitively provides "harder" positive and negative examples, allows CNC to more strongly overcome spurious correlations.

Most similar to our approach are methods that first train an initial ERM model with the class labels as a way to identify data points belonging to minority groups, and subsequently train an additional model with greater emphasis on the estimated minority groups. Sohoni et al. (2020) demonstrate that even when only trained on the class labels, neural networks learn feature representations that can be clustered into groups of data exhibiting different spurious attributes. They use the resulting cluster labels as estimated group labels before running GDRO on these estimated groups. Meanwhile, Nam et al. (2020) train a pair of models, where one model minimizes a generalized cross-entropy loss (Zhang & Sabuncu, 2018), such that the datapoints this model classifies incorrectly largely correspond to those in the minority group. They then train the other model on the same data but upweight the minority-group-estimated points. While they interweave training of the biased and robust model, Liu et al. (2021) instead train one model first with a shortened training time (but the standard cross-entropy objective), and show that then upsampling the incorrect data points and training another model with ERM can yield higher worst-group accuracy. Creager et al. (2021) propose Environment Inference for Invariant Learning (EIIL), which first trains an ERM model, and then softly assign the training data into groups under which the initial trained ERM model would maximally violate the invariant risk minimization (IRM) objective. In particular, the IRM objective is maximally satisfied if a model's optimal classifier is the same across groups (Arjovsky et al., 2019), and EIIL groups are inferred such that the initial ERM model's representations exhibit maximum variance within each group. Creager et al. (2021) then runs GDRO with these groups. Finally, Nagarajan et al. (2020) provides a theoretical understanding of how ERM picks up spurious features under data set imbalance. They consider a setting involve a single spurious feature that is correlated with the class label and analyze the max-margin classifier in the presence of this spurious feature.

In our work, we demonstrate that the ERM model's predictions can be leveraged to not only estimate groups and train a new model with supervised learning but with different weightings. Instead, we can specifically identify pairs of points that a contrastive model can then learn invariant features between. Our core contribution comes from rethinking the objective with a contrastive loss that more directly reduces the model's ability to learning spurious correlations.

D.2. Contrastive learning

Our method also uses contrastive learning, a simple yet powerful framework for both self-supervised (Chen et al., 2020; Oord et al., 2018; Tian et al., 2019; Song & Ermon, 2020; Sermanet et al., 2018; Hassani & Khasahmadi, 2020; Robinson et al., 2021) and supervised (Khosla et al., 2020; Gunel et al., 2021) representation learning. The core idea is to learn data representations that maximize the similarity between a given input "anchor" and distinct different views of the same input ("positives"). Frequently this also involves *contrasting* positives with "negative" data samples without any assumed relation to the anchor (Bachman et al., 2019). Core components then include some way to source multiple views, e.g. with data transformations (Chen et al., 2020), and training objectives similar to noise contrastive estimation (Gutmann & Hyvärinen, 2010; Mnih & Kavukcuoglu, 2013).

An important component of contrastive learning is the method by which appropriate positives and negatives are gathered. For sampling positives, Chen et al. (2020) show that certain data augmentations (e.g. crops and cutouts) may be more beneficial than others (e.g. Gaussian noise and Sobel filtering) when generating anchors and positives for unsupervised contrastive learning. (von Kügelgen et al., 2021) theoretically study how data augmentations help contrastive models learn core content attributes which are invariant to different observed "style changes". They propose a latent variable model for self-supervised learning. Tian et al. (2020) further study what makes good views for contrastive learning. They propose an

"InfoMin principle", where anchors and positives should share the least information necessary for the contrastive model to do well on the downstream task. For sampling negatives, Robinson et al. (2021) show that contrastive learning also benefits from using "hard" negatives, which (1) are actually a different class from the anchor (which they approximate in the unsupervised setting) and (2) embed closest to the anchor under the encoder's current data representation. Both of these approaches capture the principle that if positives are always too similar to the anchor and negatives are always too different, then contrastive learning may be inefficient at learning generalizable representations of the underlying classes.

In our work, we incorporate this principle by sampling data points with the same class label but different ERM predictions–presumably because of spurious attribute differences–as anchor and positive views, while sampling negatives from data points with different class labels but the same ERM prediction as the anchor. The anchors and positives are different enough that a trained ERM model predicted them differently, while the anchors and negatives are similar enough that the trained ERM model predicted them the same. Contrasting the above then allows us to exploit both "hard" positive and negative criteria for our downstream classification task. In Section 5.3, we show that removing this ERM-guided sampling (i.e. only sampling positives and negatives based on class information), as well as trying different negative sampling procedures, leads to substantially lower worst-group accuracy with CNC.

One limitation of our current theoretical analysis regarding the alignment loss (cf. Section 3.2) is that we require knowing the group labels to compute the RHS of equation (6) (in particular, the alignment loss). An interesting question for future work is to provide a better theoretical understanding of the alignment induced by CNC in the context of spurious correlations.

D.3. Invariant learning

Our work also shares some similarities in motivation with Invariant Risk Minimization (IRM) (Arjovsky et al., 2019), Predictive Group Invariance (PGI) (Ahmed et al., 2021), and other related works in domain-invariant learning (Krueger et al., 2020; Parascandolo et al., 2020; Ahuja et al., 2020; Creager et al., 2021). These methods aim to train models that learn a single invariant representation that is consistently optimal (e.g. with respect to classifying data) across different domains or environments. These environments can be thought of as data groups, and while traditionally methods such as IRM require that environment labels are known, recent approaches such as Environment Inference for Invariant Learning (EIIL) (Creager et al., 2021) and Predictive Group Invariance (PGI) (Ahmed et al., 2021) similarly aim to infer environments with an initial ERM model. As discussed in Appendix D.1, in EIIL, Creager et al. (2021) next train a more robust model with an invariant learning objective, similarly selecting models based on the worst-group error on the validation set. However, they train this model using IRM for Colored MNIST and GDRO for Waterbirds and CivilComments-WILDS, using the inferred environments as group labels. PGI uses EIIL to infer environments, but trains a more robust model by minimizing the KL divergence of the predicted probabilities for samples in the same class, but different groups, using the inferred environments as group labels. Thus, these approaches may similarly seek to learn similar representations for samples across groups, but do so via GDRO's reweighting, or the outputs of the model's classification layer. In contrast, CNC trains a more robust model by using contrastive learning to affect its representations more directly. In our main evaluation (c.f. Section 5.1) we show that CNC's proposed contrastive loss objective and hard sampling strategy lead to higher worst-group accuracy.

D.4. Learning representations for unsupervised domain adaptation

CNC's approach to improve model robustness via a model's hidden-layer representations bears similarity to some prior unsupervised domain adaptation (UDA) methods. In UDA, the goal is to use the source and target features, and the source labels, to transfer to the specific target domain. As introduced in Appendix C.6, to generalize beyond a single domain, one promising direction is to learn similar representations for samples from different domains. However, UDA methods assume knowledge of training sample domains or spurious attributes, whereas CNC and our other comparable methods do not. UDA methods also assume a fundamentally different data setup; the training data is divided into source and target domains, and only the source domain has labels. In our setting, we *do* have class labels for all samples available during training, but *do not* have any natural definition of source and target domains (and thus no training data domain labels). Applying UDA methods requires additional reintepretation of this setup.

Considering methods to learn desirable representations, one popular UDA approach is domain adversarial neural networks (DANN) (Ganin et al., 2016). DANN accomplishes alignment by adversarially training a model's encoder layers (the "feature extractor") to learn representations such that a separate domain classifier module cannot distinguish samples' domains from the learned representations. To preserve class information, they train a classifier module on top of the feature extractor jointly with a cross-entropy loss. CNC's process for aligning representations is more simple. We do not rely on

training separate modules with conflicting objectives to accomplish alignment; instead the supervised contrastive loss with CNC's sampling procedure encourages learning representations that are both separable across classes and aligned within each class. We thus avoid additional training parameters and optimization issues associated with minimax-based adversarial training (Arjovsky & Bottou, 2017). Instead of relying on a the domain classifier's output, we can train single model to align representations by minimizing the cosine similarity between anchor and positive samples.

D.5. Learning representations for domain generalization

CNC also bears some similarity to representation learning methods for domain generalization (DG), which also try to learn representations that generalize beyond individual groups. However, in contrast to the spurious correlations problem, DG settings often assume access to multiple "source" domains and knowledge (labels) of which group or domain each sample belongs to. They aim to generalize to a specific unseen "target" domain not present during training. Unlike the spurious correlations presented in our evaluated datasets, class distributions within domains are also not canonically as skewed in standard benchmarks such as Office (Saenko et al., 2010) and VLCS (Fang et al., 2013). The distribution shift presented in these benchmarks is also distinct from the shift encountered with spurious correlations benchmarks. With spuriously correlated data, a model may learn to rely on spurious features themselves to obtain high average classification accuracy on the training set (e.g., solely relying on background features to classify bird type). At test time, the model may make confident predictions solely based on the absence or presence of these spurious features. In contrast, in DG (and UDA) settings, the domain-specific features themselves do not correlate with classes. A model may learn to rely on domain-specific "style" features *in conjunction* with other "content-based" features to classify images during training. The absence of the former style features in new domains may decrease performance, but not because the model has latched on specific spurious features to dictate its predictions. The two tasks thus differ via different model behaviors and failure modes.

Methods to tackle DG via representation alignment include the invariant learning methods discussed in Appendix D.3. Additionally, (Li et al., 2018) propose *maximum mean discrepancy* MMD adversarial autoencoder (MMD-AAE). To align representations between domains, MMD-AAE (1) trains an autoencoder, (2) uses MMD maximum applied at the bottleneck hidden-layer to match representations across domains, and (3) applies an adversarial discriminator network to match these representations with a Laplace distribution (to encourage more sparse hidden representations). This matching is also not conditioned on the sample classes; an additional classifier head is applied to preserve class-specific information. CNC is also simpler than MMD-AAE, only using the normalized dot product of a single classifier's last hidden-layer representations. Via contrastive learning, CNC also critically also aims to only align representations with the same class but different ERM-inferred groups, while pushing apart samples with different classes but the same ERM-inferred groups. By paying attention to the classes, we directly encourage a model to ignore group-specific information which confused the initial ERM model but that does not discriminate between classes.

E. Additional experimental details

We first further describe our evaluation benchmarks in Appendix E.1. We next provide further details on how we calculate the reported metrics and the experimental hyperparameters of our main results in Appendix E.1. For all methods, following prior work (Liu et al., 2021; Sohoni et al., 2020; Nam et al., 2020; Sagawa et al., 2019; Creager et al., 2021) we report the test set worst-group and average accuracies from models selected through hyperparameter tuning for the best validation set worst-group accuracy. While different methods have different numbers of tunable hyperparameters, we try to keep the number of validation queries as close as possible while tuning for fair comparison.

E.1. Dataset details

Colored MNIST (CMNIST*). We evaluate with a version of the Colored MNIST dataset proposed in Arjovsky et al. (2019). The goal is to classify MNIST digits belonging to one of 5 classes $\mathcal{Y} = \{(0, 1), (2, 3), (4, 5), (6, 7), (8, 9)\}$, and treat color as the spurious attribute. In the training data, we color p_{corr} of each class's datapoints with an associated color a, and color the rest randomly. If p_{corr} is high, trained ERM models fail to classify digits that are not the associated color. We pick a from uniformly interspersed intervals of the `hsv` colormap, e.g. 0 and 1 digits may be spurious correlated with the color red (`#ff0000`), while 8 and 9 digits may be spuriously correlated with purple (`#ff0018`). The full set of colors in class order are $\mathcal{A} = \{$`#ff0000`, `#85ff00`, `#00fff3`, `#6e00ff`, `#ff0018`$\}$ (see Fig. 2). For validation and test data, we color each datapoint randomly with a color $a \in \mathcal{A}$. We use the default test set from MNIST, and allocate 80%-20% of the default MNIST training set to the training and validation sets. For main results, we set $p_{\text{corr}} = 0.995$.

Waterbirds. We evaluate with the Waterbirds dataset, which was introduced as a standard spurious correlations benchmark in Sagawa et al. (2019). In this dataset, masked out images of birds from the CUB dataset (Wah et al., 2011) are pasted on backgrounds from the Places dataset (Zhou et al., 2017). Bird images are labeled either as waterbirds or landbirds; background either depicts water or land. From CUB, waterbirds consist of seabirds (ablatross, auklet, cormorant, frigatebird, fulmar, gull, jaeger, kittiwake, pelican, puffin, tern) and waterfowl (gadwell, grebe, mallard, merganser, guillemot, Pacific loon). All other birds are landbirds. From Places, water backgrounds consist of ocean and natural lake classes, while land backgrounds consist of bamboo forest and broadleaf forest classes.

The goal is to classify the foreground bird as $\mathcal{Y} = \{\text{waterbird, landbird}\}$, where there is spurious background attribute $\mathcal{A} = \{\text{water background, land background}\}$. We use the default training, validation, and test splits (Sagawa et al., 2019), where in the training data 95% of waterbirds appear with water backgrounds and 95% of landbirds appear with land backgrounds. Trained ERM models then have trouble classifying waterbirds with land backgrounds and landbirds with water backgrounds. For validation and test sets, water and land backgrounds are evenly split among landbirds and waterbirds.

CelebA. We evaluate with the CelebA spurious correlations benchmark introduced in Sagawa et al. (2019). The goal is to classify celebrities' hair color $\mathcal{Y} = \{\text{blond, not blond}\}$, which is spuriously correlated with the celebrity's identified gender $\mathcal{A} = \{\text{male, female}\}$. We use the same training, validation, test splits as in Sagawa et al. (2019). Only 6% of blond celebrities are male; trained ERM models perform poorly on this group.

CivilComments-WILDS. We evaluate with the CivilComments-WILDS dataset from Koh et al. (2021), derived from the Jigsaw dataset from Borkan et al. (2019). Each datapoint is a real online comment curated from the Civil Comments platform, a commenting plugin for independent news sites. For classes, each comment is labeled as either toxic or not toxic. For spurious attributes, each comment is also labeled with the demographic identities {male, female, LGBTQ, Christian, Muslim, other religions, Black, White} mentioned; multiple identities may be mentioned per comment.

The goal is to classify the comment $\mathcal{Y} = \{\text{toxic, not toxic}\}$. As in Koh et al. (2021), we evaluate with $\mathcal{A} = \{\text{male, female, LGBTQ, Christian, Muslim, other religions, Black, White}\}$. There are then 16 total groups corresponding to (toxic, identity) and (not toxic, identity) for each identity. Groups may overlap; a datapoint falls in a group if it mentions the identity. We use the default data splits (Koh et al., 2021). In Table 11, we list the percentage of toxic comments for each identity based on the groups. Trained ERM models in particular perform less well on the rarer toxic groups.

Table 11. Percent of toxic comments for each identity in the CivilComments-WILDS training set.

Identity	male	female	LGBTQ	Christian	Muslim	other religions	Black	White
% toxic	14.9	13.7	26.9	9.1	22.4	15.3	31.4	28.0

E.2. Implementation details

E.2.1. Reported metrics

Main results. For the CMNIST*, Waterbirds, and CelebA datasets, we run CNC with three different seeds, and report the average worst-group accuracy over these three trials in Table 1. As we use the same baselines and comparable methods as Liu et al. (2021), we referenced their main results for the reported numbers, which did not have standard deviations or error bars reported. For CivilComments-WILDS, due to time and compute constraints we only reported one run. We note that CMNIST* here is extremely challenging, as minority groups together only make up 0.5% of the training set. This severe imbalance explains the very poor worst-group performance of ERM (and other methods that fail to sufficiently remediate the issue).

Estimated mutual information. We give further details for calculating the representation metric introduced in Sec. 3. As a reminder, we report both alignment and estimated mutual information metrics to quantify how dependent a model's learned representations are on the class labels versus the spurious attributes, and compute both metrics on the representations $Z = \{f_{\text{enc}}(x)\}$ over all test set data points x. Then to supplement the alignment loss calculation in Sec. 3, we also estimate $I(Y; Z)$ and $I(A; Z)$, the mutual information between the model's data representations and the class labels and spurious attribute labels respectively.

To first estimate mutual information with Y, we first approximate $p(y \mid z)$ by fitting a multinomial logistic regression model

over all representations Z to classify y. With the empirical class label distribution $p(y)$, we compute:

$$\hat{I}(Y;Z) = \frac{1}{|Z|} \sum_{z \in Z} \sum_{y \in Y} p(y \mid z) \log \frac{p(y \mid z)}{p(y)} \tag{18}$$

We do the same but substitute the spurious attributes a for y to compute $\hat{I}(A;Z)$.

E.2.2. STAGE 1 ERM TRAINING DETAILS

We describe the model selection criterion, architecture, and training hyperparameters for the initial ERM model in our method. To select this model, recall that we first train an ERM model to predict the class labels, as the model may also learn dependencies on the spurious attributes. Because we then use the model's predictions on the training data to infer samples with different spurious attribute values but the same class label, we prefer an initial ERM model that better learns this spurious dependency, and importantly also does not overfit to the training data. Inspired by the results in prior work (Sohoni et al., 2020; Liu et al., 2021), we then explored using either a standard ERM model, one with high ℓ-2 regularization (`weight decay = 1`), or one only trained on a few number of epochs. To select among these, because the validation data has both class labels and spurious attributes, we choose the model with the largest gap between worst-group and average accuracy on the validation set. . We detail the ERM architecture and hyperparameters for each dataset below:

Colored MNIST. We use the LeNet-5 CNN architecture in the `pytorch` image classification tutorial. We train with SGD, few epochs $E = 5$, SGD, learning rate 1e-3, batch size 32, default weight decay 5e-4, and momentum 0.9.

Waterbirds. We use the `torchvision` implementation of ResNet-50 with pretrained weights from ImageNet as in Sagawa et al. (2019). Also as in (Sagawa et al., 2019), we train with SGD, default epochs $E = 300$, learning rate 1e-3, batch size 128, and momentum 0.9. However we use high weight decay 1.0.

CelebA. We also use the `torchvision` ImageNet-pretrained ResNet-50 and default hyperparameters from Sagawa et al. (2019) but with high weight decay: we train with SGD, default epochs $E = 50$, learning rate 1e-4, batch size 128, momentum 0.9, and high weight decay 0.1.

CivilComments-WILDS. We use the HuggingFace (`pytorch-transformers`) implementation of BERT with pre-trained weights and number of tokens capped at 300 as in Koh et al. (2021). As in Liu et al. (2021), with other hyperparameters set to their defaults (Koh et al., 2021) we tune between using the AdamW optimizer with learning rate 1e-5 and SGD with learning rate 1e-5, momentum 0.9, and the PyTorch `ReduceLROnPlateau` learning rate scheduler. Based on our criterion, we use SGD, few number of epochs $E = 2$, learning rate 1e-5, batch size 16, default weight decay 1e-2, and momentum 0.9.

E.2.3. CONTRASTIVE BATCH SAMPLING DETAILS

We provide further details related to collecting predictions from the trained ERM models, and the number of positives and negatives that determine the contrastive batch size.

ERM model prediction. To collect trained ERM model predictions on the training data, we explored two approaches: (1) using the actual predictions, i.e. the argmax for each classifier layer output vector, and (2) clustering the representations, or the last hidden-layer outputs, and assigning a cluster-specific label to each data point in one cluster. This latter approach is inspired by Sohoni et al. (2020), and we similarly note that ERM models trained to predict class labels in spuriously correlated data may learn data representations that are clusterable by spurious attribute. As a viable alternative to collecting the "actual" predictions of the trained ERM model on the training data, with C classes, we can then cluster these representations into C clusters, assign the same class label only to each data point in the same cluster, and choose the label-cluster assignment that leads to the highest accuracy on the training data. We also follow their procedure to first apply UMAP dimensionality reduction to 2 UMAP components, before clustering with K-means or GMM (Sohoni et al., 2020). To choose between all approaches, we selected the procedure that lead to highest worst-group accuracy on the validation data after the second-stage of training. While this cluster-based prediction approach was chosen as a computationally efficient heuristic, we found that in practice it either lead to comparable or better final worst-group accuracy on the validation set. To better understand this, as a preliminary result we found that when visualizing the validation set predictions with the Waterbirds dataset, the cluster-based predictions captured the actual spurious attributes better than the classifier layer predictions (Fig. 11). We defer additional discussion to Sohoni et al. (2020) and leave further analysis to future work.

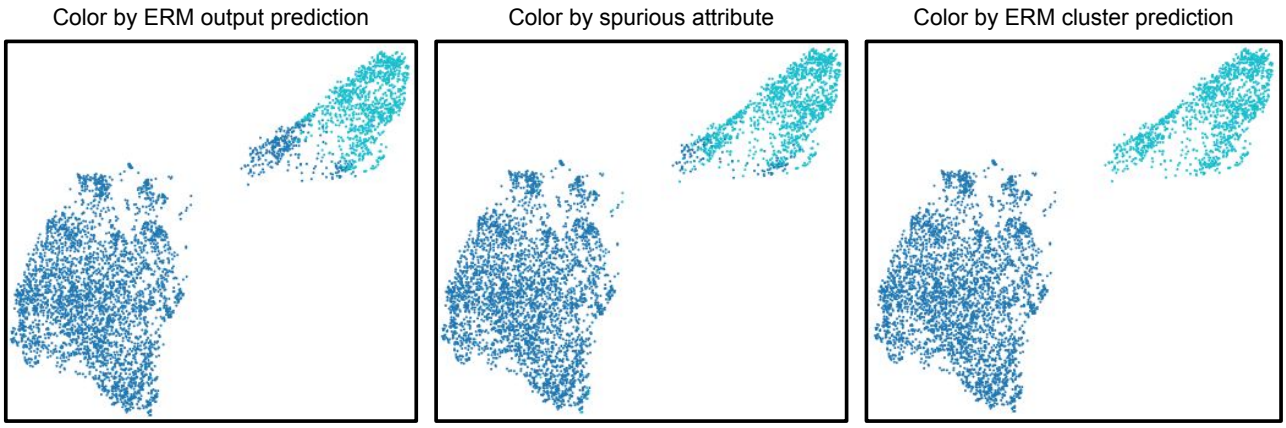


Figure 11. UMAP visualization of ERM data representations for the Waterbirds training data. We visualize the last hidden layer outputs for a trained ERM ResNet-50 model given training samples from Waterbirds, coloring by either the ERM model's "standard" predictions, the actual spurious attribute values (included here just for analysis), and predictions computed by clustering the representations as described above. Clustering-based predictions more closely align with the actual spurious attributes than the ERM model outputs.

Number of positives and negatives per batch. One additional difference between our work and prior contrastive learning methods (Chen et al., 2020; Khosla et al., 2020) is that we specifically construct our contrastive batches by sampling anchors, positives, and negatives first. This is different from the standard procedure of randomly dividing the training data into batches first, and then assigning the anchor, positive, and negative roles to each datapoint in a given batch. As a result, we introduce the number of positives M and the number of negatives N as two hyperparameters that primarily influence the size of each contrastive batch (with number of additional anchors and negatives also following M and N with two-sided batches). To maximize the number of positive and negative comparisons, as a default we set M and N to be the maximum number of positives and negatives that fit the sampling criteria specified under Algorithm 2 that fit in memory. In Appendix E.2.4, for each dataset we detail the ERM prediction method and number of positives and negatives sampled in each batch.

E.2.4. Stage 2 contrastive model training details

In this section we describe the model architectures and training hyperparameters used for training the second model of our procedure, corresponding the reported worst-group and average test set results in Table 1. In this second stage, we train a new model with the same architecture as the initial ERM model, but now with a contrastive loss and batches sampled based on the initial ERM model's predictions. We report test set worst-group and average accuracies from models selected with hyperparameter tuning and early stopping based on the highest validation set worst-group accuracy. For all datasets, we sample contrastive batches using the clustering-based predictions of the initial ERM model. Each batch size specified here is also a direct function of the number of positives and negatives: $2M + 2N$.

Colored MNIST. We train a LeNet-5 CNN. For CNC, we use $M = 32$, $N = 32$, batch size 128, temperature $\tau = 0.05$, contrastive weight $\lambda = 0.75$, SGD optimizer, learning rate 1e-3, momentum 0.9, and weight decay 1e-4. We train for 3 epochs, and use gradient accumulation to update model parameters every 32 batches.

Waterbirds. We train a ResNet-50 CNN with pretrained ImageNet weights. For CNC, we use $M = 17$, $N = 17$, batch size 68, temperature $\tau = 0.1$, contrastive weight $\lambda = 0.75$, SGD optimizer, learning rate 1e-4, momentum 0.9, weight decay 1e-3. We train for 5 epochs, and use gradient accumulation to update model parameters every 32 batches.

CelebA. We train a ResNet-50 CNN with pretrained ImageNet weights. For CNC, we use $M = 64$, $N = 64$, batch size 256, temperature $\tau = 0.05$, contrastive weight $\lambda = 0.75$, SGD optimizer, learning rate 1e-5, momentum 0.9, and weight decay 1e-1. We train for 15 epochs, and use gradient accumulation to update model parameters every 32 batches.

CivilComments-WILDS. We train a BERT model with pretrained weights and max number of tokens 300. For CNC, we use $M = 16$, $N = 16$, batch size 64, temperature $\tau = 0.1$, contrastive weight $\lambda = 0.75$, AdamW optimizer, learning rate 1e-4, weight decay 1e-2, and clipped gradient norms. We train for 10 epochs, and use gradient accumulation to update weights every 128 batches.

E.2.5. COMPARISON METHOD TRAINING DETAILS

As reported in the main results (Table 1) we compare CNC with the ERM and Group DRO baselines, as well as robust training methods that do not require spurious attribute labels for the training data: GEORGE (Levy et al., 2020), Learning from Failure (LfF) (Levy et al., 2020), Predictive Group Invariance (PGI) (Ahmed et al., 2021), Contrastive Input Morphing (CIM) (Taghanaki et al., 2021), Environment Inference for Invariant Learning (EIIL) (Creager et al., 2021), and Just Train Twice (JTT) (Liu et al., 2021). For each dataset, we use the same model architecture for all methods. For the Waterbirds, CelebA, and CivilComments-WILDS datasets, we report the worst-group and average accuracies reported in Liu et al. (2021) for ERM, LfF, and JTT. For GEORGE, we report the accuracies reported in Sohoni et al. (2020). For CIM, we report results from Waterbirds and CelebA from (Taghanaki et al., 2021) using the CIM + variational information bottleneck implementation (Alemi et al., 2016), which achieves the best worst-group performance in their results. For EIIL, we report results from Waterbirds and CivilComments-WILDS from (Creager et al., 2021). For these hyperparameters, we defer to the original papers. For GDRO, we reproduce the results with the same optimal hyperparameters over three seeds. For PGI, we used our own implementation for all results, with details specified below. For Colored MNIST, we run implementations for GEORGE, CIM, EIIL, LfF, JTT, and GDRO, using code from their authors respectively. We include training details for our own implementations below:

Colored MNIST (CMNIST*). We run all methods for 20 epochs, reporting test set accuracies with early stopping. For JTT, we train with SGD, learning rate 1e-3, momentum 0.9, weight decay 5e-4, batch size 32. We use the same initial ERM model as CNC, with hyperparameters described in Appendix E.2.2. For upsampling we first tried constant factors $\{10, 100, 1000\}$. We also tried a resampling strategy where for all the datapoints with the same initial ERM model prediction, we upsample the incorrect points such that they equal the correct points in frequency, and found this worked the best. With $p_{\text{corr}} = 0.995$, this upsamples each incorrect point by roughly 1100. We also use this approach for the results in Fig. 7. For GDRO we use the same training hyperparameters as JTT, but without the upsampling and instead set group adjustment parameter $C = 0$. For LfF, we use the same hyperparameters as JTT, but instead of upsampling gridsearched the q parameter $\in \{0.1, 0.3, 0.5, 0.7, 0.9\}$, using $q = 0.7$. For GEORGE we train with SGD, learning rate 1e-3, momentum 0.9, weight decay 5e-4. For CIM, we use the CIM + VIB implementation. We train with SGD, learning rate 1e-3, weight decay 5e-4, β parameter 10, and λ parameter 1e-5. For EIIL, for environment inference we use the same initial ERM model as CNC and JTT, and update the soft environment assignment distribution with Adam optimizer, learning rate 1e-3, and 10000 steps. Following (Creager et al., 2021)'s own colored MNIST experiment, we train the second model with IRM, using learning rate 1e-2, weight decay 1e-3, penalty weight 100, and penalty annealing parameter 80.

CelebA. We also tune EIIL for CelebA. We again use the same initial ERM model as CNC, and update the soft environment assignment distribution with Adam optimizer, learning rate 1e-3, and 10000 steps. We train the second model with GDRO, using SGD, 50 epochs, learning rate 1e-5, batch size 128, weight decay 0.1, and group adjustment parameter 3.

PGI. To compare against PGI, we tried two implementations. First, we followed the PGI algorithm to first infer environments via the same mechanism as in EIIL [2], and trained a second model with the PGI objective using standard shuffled minibatches (aiming to minimize the KL divergence for samples with the same class but different inferred environment labels per batch). However, despite ample hyperparameter tuning (trying loss weighting component $\lambda \in \{0.1, 0.5, 10, 100\}$, we could not get PGI to work well (on Waterbirds, we obtained $51.0 \pm 4.9\%$ worst-group accuracy and $79.6 \pm 2.6\%$ average accuracy). We hypothesize this is due to the strong spurious correlations in our datasets: while (Ahmed et al., 2021) only considers datasets where 20% of the training samples do not exhibit a dominant correlation and fall under minority groups. Our evaluation benchmarks are more difficult due to stronger spurious correlations, e.g., in Waterbirds only 5% of samples do not exhibit the dominant correlation; similarly only 7% of training samples lie in the smallest group in CelebA.

We then tried a more balanced batch variation. Instead of using randomly shuffled minibatches, we used the PGI environment inference labels to sample batches similarly to how CNC uses the stage 1 ERM model predictions to sample batches. We construct batches by specifying the same number of "anchors", "positives", and "negatives" as in CNC, and sample batches where anchors and positives are samples with the same class, but different inferred environments. Anchors and negatives are samples in the same inferred environment, but with different classes. We then trained a second model with the PGI criterion with these modified batches.

In Section E.2.6, we include our sweeps for both the method-specific and general hyperparameters.

Comparison limitations. One limitation of our comparison is that because for each dataset we sample new contrastive batches which could repeat certain datapoints, the number of total batches per epoch changes. For example, 50 epochs

training the second model in CNC does not necessarily lead to the same total number of training batches as 50 epochs training with ERM, even if they use the same batch size. However, we note that the numbers we compare against from Liu et al. (2021) are reported with early stopping. In this sense we are comparing the best possible worst-group accuracies obtained by the methods, not the highest worst-group accuracy achieved within a limited number of training batches. We also found that although in general the time to complete one epoch takes much longer with CNC, CNC requires fewer overall training epochs for all but the CivilComments-WILDS dataset to obtain the highest reported accuracy.

E.2.6. HYPERPARAMETER SWEEPS

To fairly compare with previous methods (Liu et al., 2021; Creager et al., 2021; Taghanaki et al., 2021), we use the same evaluation scheme (selecting models based on worst-group validation error), and sweep over a consistent number of hyperparameters, i.e. number of validation set queries. We set this number for CNC to be a comparable number of queries that is reported in prior works. We break this down into method-specific (e.g. contrastive temperature in CNC, upweighting factor in JTT), and shared (e.g. learning rate) hyperparameter categories.

Method-specific. For CNC, we tune three method-specific hyperparameters: contrastive loss temperature (Eq. 3), contrastive weight (Eq. 7), and gradient accumulation steps values as in Table 12.

Table 12. Method-specific hyperparameters for CNC.

Hyperparameter	Dataset	Values
Temperature (τ)	All	$\{0.05, 0.1\}$
Contrastive Weight (λ)	All	$\{0.5, 0.75\}$
Gradient Accumulation Steps	CMNIST*, Waterbirds, CelebA	$\{32, 64\}$
	CivilComments-WILDS	$\{32, 64, 128\}$

For JTT, the reported results and our CMNIST* implementation are tuned over the following hyperparameters in Table 13.

Table 13. Method-specific hyperparameters for JTT.

Hyperparameter	Dataset	Values
Stage 1 Training Epochs	Waterbirds	$\{40, 50, 60\}$
	CMNIST*, CelebA, CivilComments-WILDS	$\{1, 2\}$
Upweighting Factor	CMNIST*	$\{10, 100, 1000, 1100\}$
	Waterbirds, CelebA	$\{20, 50, 100\}$
	CivilComments-WILDS	$\{4, 5, 6\}$

For EIIL, our CMNIST* and CelebA implementations are tuned over hyperparameters reported in Table 14. (Creager et al., 2021) report that they allow up to 20 evaluations with different hyperparameters for Waterbirds and CivilComments-WILDS. When using GDRO as the second stage model, they also report using the same hyperparameters as the GDRO baseline for Waterbirds. We do the same for our evaluation on CelebA. This amounts to primarily tuning the first stage environment inference learning rate and number of updating steps for CMNIST* and CelebA, and the penalty annealing iterations and penalty weight for the IRM second stage model for CMNIST*.

Table 14. Method-specific hyperparameters for EIIL.

Hyperparameter	Dataset	Values
Environment Inference Learning Rate	CMNIST*, CelebA	{1e-1, 1e-2, 1e-3}
Environment Inference Update Steps	CMNIST*, CelebA	{10000, 20000}
IRM Penalty Weight	CMNIST*	{0.1, 10, 1000, 1e5}
IRM Penalty Annealing Iterations	CMNIST*	{10, 50, 80}
GDRO Group Adjustment	CelebA	{0, 2, 3}

For CIM, our CMNIST* implementation uses CIM + VIB, and is tuned over the β VIB parameter (Alemi et al., 2016) and contrastive weighting parameter λ for CIM in Table 15. (Taghanaki et al., 2021) report tuning over a range of values within [1e-5, 1] for λ on the CelebA and Waterbirds datasets.

Table 15. Method-specific hyperparameters for CIM.

Hyperparameter	Dataset	Values
CIM λ	CMNIST*	{0.01, 0.05, 0.1}
VIB β	CMNIST*	{1e-5, 1e-3, 1e-1, 10}

For PGI, we tune λ with the same environment inference parameters as used in EIIL (Table 14). Fixing these parameters to infer environments, we tuned the λ component for training the robust model across $\lambda \in \{0.1, 0.5, 10, 100\}$.

Shared. For all datasets, we use the same optimizer and momentum (if applicable) as reported in the JTT paper. Table 16 contains the data-specific shared hyperparameter values tried.

Table 16. Shared hyperparameters

	CMNIST*	Waterbirds	CelebA	CivilComments-WILDS
Learning Rate	{1e-4, 1e-3, 1e-2}	{1e-4, 1e-3}	{1e-5, 1e-4}	{1e-5, 1e-4}
Weight Decay	{1e-4, 5e-4}	{1e-4, 1e-3}	{1e-2, 1e-1}	{1e-2}

E.3. CNC compute resources and training time

All experiments for CMNIST*, Waterbirds, and CelebA were run on a machine with 14 CPU cores and a single NVIDIA Tesla P100 GPU. Experiments for CivilComments-WILDS were run on an Amazon EC2 instance with eight CPUs and one NVIDIA Tesla V100 GPU.

Regarding runtime, one limitation with the current implementation of CNC is its comparatively longer training time compared to methods such as standard ERM. This is both a result of training an initial ERM model in the first stage, and training another model with contrastive learning in the second stage. In Table 17 we report both how long it takes to train the initial ERM model and long it takes to complete one contrastive training epoch on each dataset. We observe that while in some cases training the initial ERM model is negligible, especially if we employ training with only a few epochs to prevent memorization (for Colored MNIST it takes roughly two minutes to obtain a sufficient initial ERM model), it takes roughly 1.5 and 3 hours to train the high regularization initial models used for Waterbirds and CelebA. While these hurdles are shared by all methods that train an initial ERM model, we find that the second stage of CNC occupies the bulk of training time. Prior work has shown that contrastive learning typically requires longer training times and converges more slowly than supervised learning (Chen et al., 2020). We also observe this in our work.

We note however that because we sample batches based on the ERM model's predictions, the contrastive training duration is limited by how many datapoints the initial ERM model predicts incorrectly. In moderately sized datasets with very few datapoints in minority groups, (e.g. Waterbirds, which has roughly 4794 training points and only 56 datapoints in its smallest group), the total time it takes to train CNC is on par with ERM. Additionally, other methods such as additional hard negative mining (Robinson et al., 2021) have been shown to improve the efficiency of contrastive learning, and we can incorporate these components to speed up training time as well.

Table 17. CNC Average total training time for first and second stages of CNC

Dataset	CMNIST*	Waterbirds	CelebA	CivilComments-WILDS
Stage 1 ERM train time	2 min.	1.5 hrs	3 hrs	3.1 hrs
Stage 2 CNC train time	1.2 hrs	1.8 hrs	32.2 hrs	37.6 hrs

F. Visualization of learned data representations

As in Fig. 6, we visualize and compare the learned representations of test set samples from models trained with ERM, JTT, and CNC in Fig. 12. Compared to ERM models, both JTT and CNC models learn representations that better depict dependencies on the class labels. However, especially with the Waterbirds and CelebA datasets, CNC model representations more clearly depict dependencies only on the class label, as opposed to JTT models which also show some organization by the spurious attribute still.

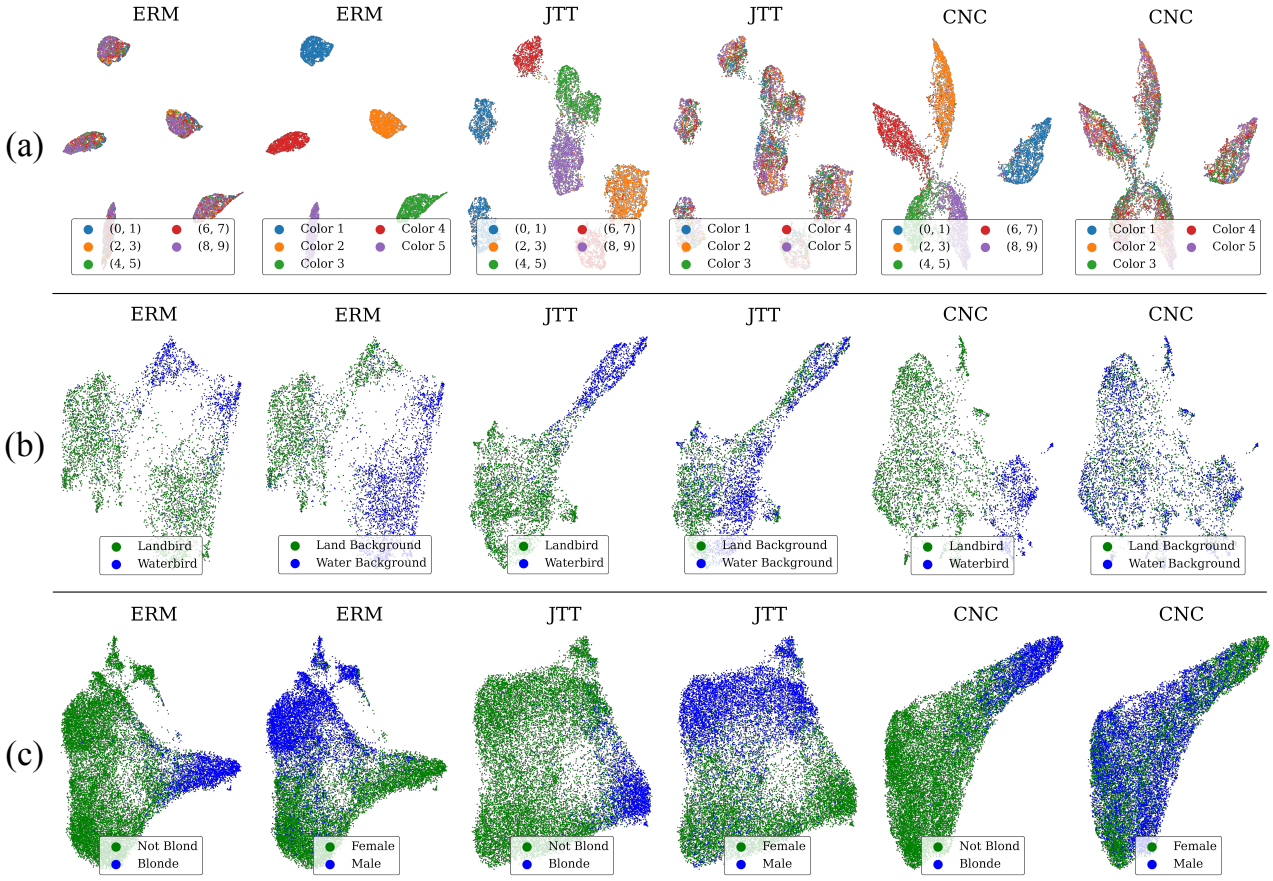


Figure 12. UMAP visualizations of learned representations for Colored MNIST (a), Waterbirds (b), and CelebA (c). We color data points based on the class label (left) and spurious attribute (right). Most consistently across datasets, CNC representations exhibit dependence and separability by the class label but not the spurious attribute, suggesting that they best learn features which only help classify class labels.

G. Additional GradCAM visualizations

On the next two pages, we include additional GradCAM visualizations depicting saliency maps for samples from each group in the Waterbirds and CelebA datasets. Warmer colors denote higher saliency, suggesting that the model considered these pixels more important in making the final classification as measured by gradient activations. For both datasets, we compare maps from models trained with ERM, the next most competitive method for worst-group accuracy JTT, and CNC. CNC models most consistently measure highest saliency with pixels directly associated with class labels and not spurious attributes.

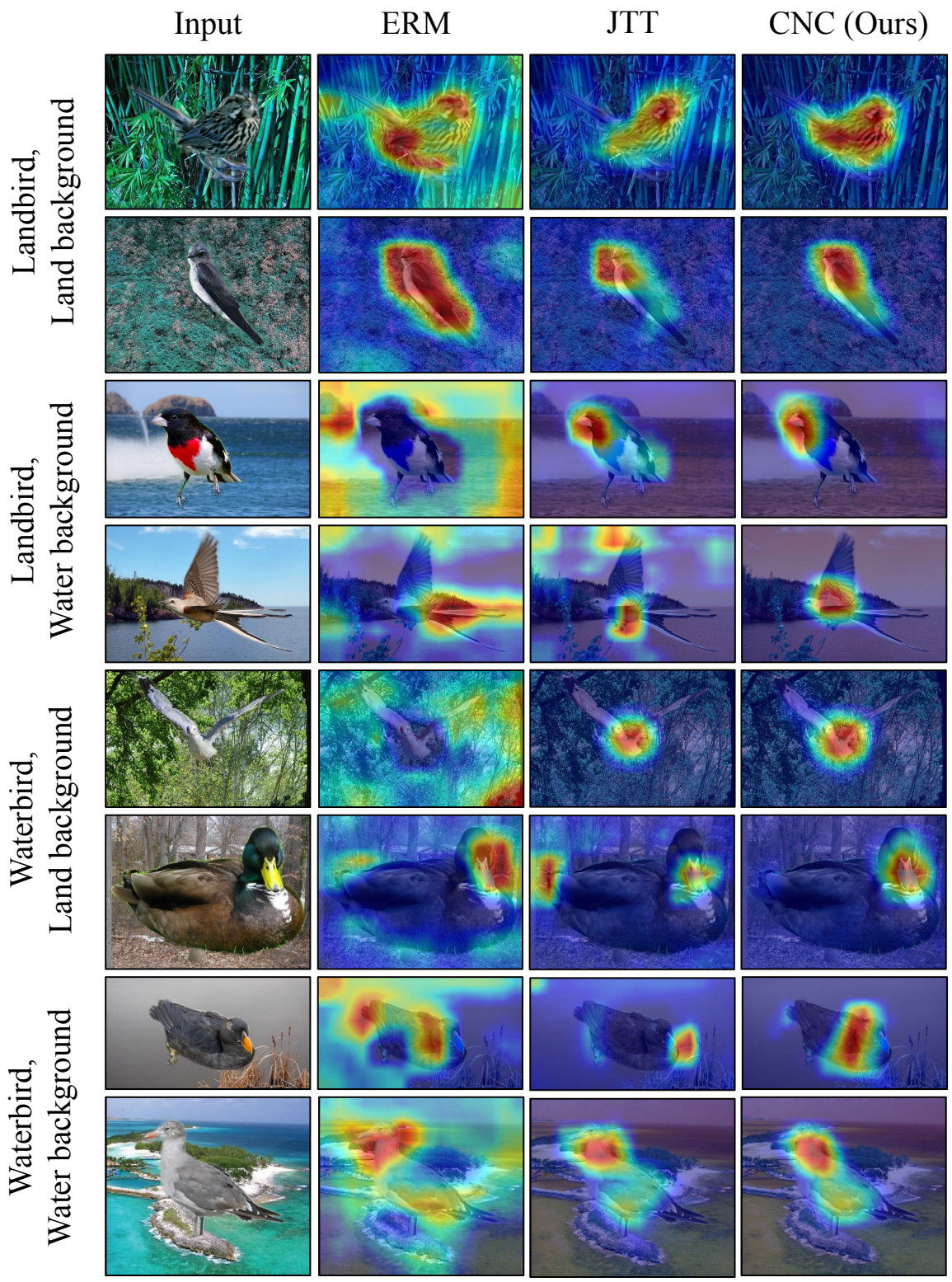


Figure 13. Additional GradCAM visualizations for the Waterbirds dataset. We use GradCAM to visualize the "salient" observed features used to classify images by bird type for models trained with ERM, JTT, and CNC. ERM models output higher salience for spurious background attribute pixels, sometimes almost exclusively. JTT and CNC models correct for this, with CNC better exclusively focusing on bird pixels.

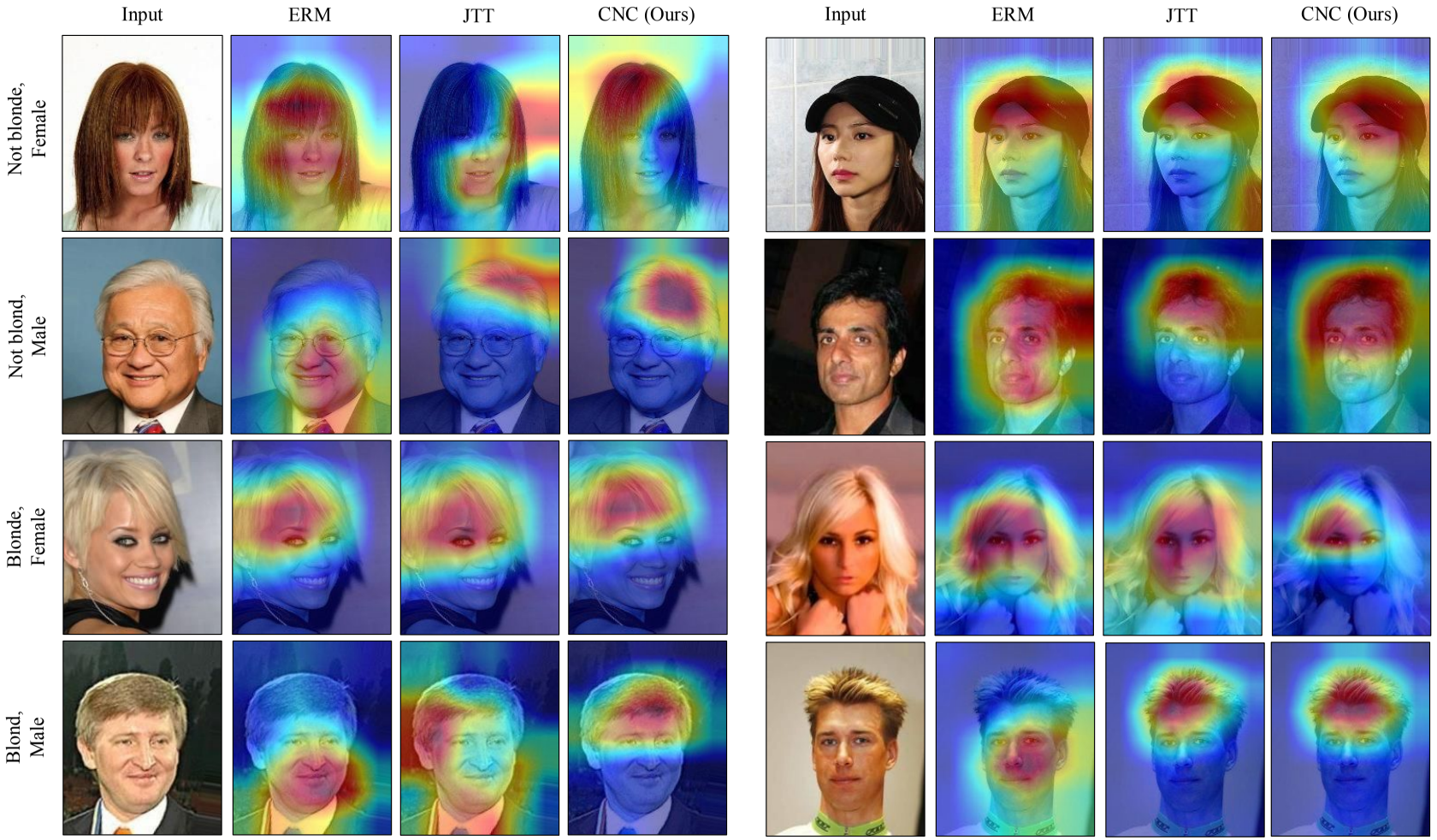


Figure 14. Additional GradCAM visualizations for the CelebA dataset. For models trained with ERM, JTT, and CNC, we use GradCAM to visualize the "salient" observed features used to classify whether a celebrity has blond(e) hair. ERM models interesting also "ignore" the actual hair pixels in favor of other pixels, presumably associated with the spurious gender attribute. In contrast, GradCAMs for JTT and CNC models usually depict higher salience for regions that at least include hair pixels. CNC models most consistently do so.