Connect, Not Collapse: Explaining Contrastive Learning for Unsupervised Domain Adaptation

Kendrick Shen [*1] Robbie Jones [*1] Ananya Kumar [*1] Sang Michael Xie [*1] Jeff Z. HaoChen [1]
Tengyu Ma [1] Percy Liang [1]

Abstract

We consider unsupervised domain adaptation (UDA), where labeled data from a source domain (e.g., photos) and unlabeled data from a target domain (e.g., sketches) are used to learn a classifier for the target domain. Conventional UDA methods (e.g., domain adversarial training) learn domain-invariant features to generalize from the source domain to the target domain. In this paper, we show that contrastive pre-training, which learns features on unlabeled source and target data and then fine-tunes on labeled source data, is competitive with strong UDA methods. However, we find that contrastive pre-training does not learn domain-invariant features, diverging from conventional UDA intuitions. We show theoretically that contrastive pre-training can learn features that vary subtantially across domains but still generalize to the target domain, by disentangling domain and class information. We empirically validate our theory on benchmark vision datasets.

1. Introduction

Machine learning models can perform poorly when the train and test data are drawn from different distributions, which is especially troublesome for performance-critical applications such as image recognition for self-driving cars (Yu et al., 2020; Sun et al., 2020) or medical image diagnosis (AlBadawy et al., 2018; Dai & Gool, 2018). In this work, we study the unsupervised domain adaptation (UDA) setting where we have access to labeled data from a source domain and unlabeled data from a target domain, and the goal is to get high accuracy on the target domain.

Conventional algorithms for UDA aim to learn domain-invariant features (Tzeng et al., 2014; Ganin et al., 2016; Tzeng et al., 2017; Shu et al., 2018; Sun et al., 2019)—intuitively, if the distributions over features for the source and target domains are indistinguishable and the accuracy is high on the source, then the accuracy should be high on the target as well. This is supported by theoretical notions such as $\mathcal{H}\Delta\mathcal{H}$-divergence, which measures the distinguishability of source and target feature spaces (Ben-David et al., 2010).

In this paper, we find that a surprisingly simple and effective method for UDA is out-of-the-box contrastive pre-training on source and target unlabeled data, followed by fine-tuning on source labeled data. In our experiments, contrastive pre-training obtains comparable or better results to strong UDA methods based on domain adversarial neural networks (Ganin et al., 2016; Shu et al., 2018) and self-training (Prabhu et al., 2021) on visual adaptation benchmarks including DomainNet, BREEDS Living-17, BREEDS Entity-30, and STL-10→CIFAR-10 (results in Table 2).

However, we show that contrastive pre-training diverges from conventional UDA intuitions and learns features that are *easily separable* between domains; for example in the learned feature space in DomainNet, we can predict the domain of an image with only 8% error, which is much lower than in the DANN feature space (14%) or in the input space (27%)—see Table 1 in Section 4. In fact, in the contrastive pre-trained feature space for DomainNet, it is as easy to distinguish betweeen two domains as it is to distinguish between two classes.

How does contrastive pre-training learn features that generalize to the target domain without domain invariance? To theoretically analyze contrastive pre-training, we consider a graph (see Figure 1) where input examples are nodes and the edge weight between two inputs is the probability that they are generated by sampling two augmentations of the same original input. By considering the total edge weight between two class-domain pairs (e.g., all edges connecting a clock photo to a butterfly sketch), we can define *class-domain connectivities*. These connectivities describe how much overlap to expect amongst the randomly augmented inputs from two class-domain pairs. Our key assumption intuitively states that augmentations of clock photos should be more similar to clock sketches and butterfly photos than to butterfly sketches.

[*] Equal contribution [1] Department of Computer Science, Stanford University. Correspondence to: Kendrick Shen, Robbie Jones, Ananya Kumar, Sang Michael Xie <{kshen6, rmjones, ananya, xie}@cs.stanford.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

Concretely, we assume that that both (1) the probability that augmentation changes the domain only, keeping the class constant, and (2) the probability that augmentation changes the class only, keeping the domain constant, should be higher than (3) the probability that augmentation changes both domain and class. Under a stochastic block model (a standard randomized model for the edges of the graph), we prove that contrastive pre-training achieves good target accuracy without domain invariance—by learning a "disentangled" feature space that is simultaneously predictive of domain and class (Theorem 1). Importantly, in our analysis the source and target domains can be very far apart, and the features can be easily distinguished (Proposition 2).

We empirically validate our theory on benchmark UDA datasets. We first heuristically estimate the connectivities between different class-domain pairs (e.g., clock photos and butterfly sketches) on Living-17 (Santurkar et al., 2020) and DomainNet (12 domain pairs) (Peng et al., 2019), showing that they satisfy our theoretical condition for contrastive pre-training to learn transferable features. Next, we find that the target accuracy of contrastive pre-training on DomainNet can be well-explained ($R^2 = 0.78$) using the terms that appear in our theory: the ratios of the estimated connectivities. Furthermore, pre-training on a selected subset of the data to adversely change the connectivities consistently underperforms pre-training on randomly selected subsets of the same size (resulting in 4%-16% drops in target accuracy). Finally, we give evidence that contrastive pre-training learns features that contain class and domain information along approximately different dimensions, allowing the features to be transferable across domains without requiring domain invariance. Our results provide a first step for theoretically understanding contrastive pre-training for UDA, which can be used to improve data augmentation and selection strategies.

2. Related work

Conventional domain adaptation methods bring source and target domains together. Ben-David et al. (2010) prove generalization bounds for domain adaptation that rely on small $\mathcal{H}\Delta\mathcal{H}$-divergence between the source and target (i.e., distributions of $\phi(x)$ for both domains are similar and therefore difficult to distinguish). Many methods such as domain adversarial training (Tzeng et al., 2014; Ganin et al., 2016; Tzeng et al., 2017) and moment matching (Peng et al., 2019) have objectives that aim to minimize $\mathcal{H}\Delta\mathcal{H}$-divergence between the source and target features. More recent works propose alternate measures of domain divergence to minimize (Li et al., 2021), propose dynamically weighting domain divergence and classification losses (Xiao & Zhang, 2021), and improve on prior work that uses optimal transport to bring the domains together (Li et al., 2020).

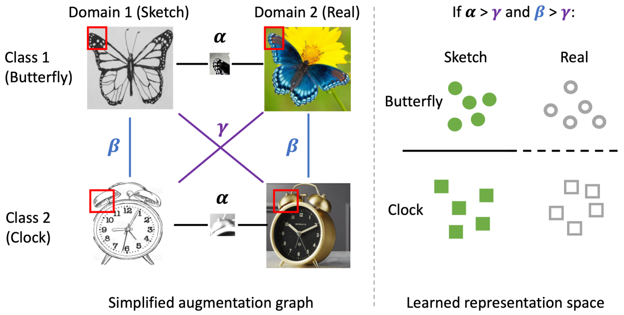

Figure 1: **(Left)** Edges between class-domain pairs indicate how "connected" they are by augmentations (e.g., cropping, colorization). For example, α denotes the probability that augmentations connect examples of the same class across different domains. Small crop sizes (e.g., the red rectangles) can increase the connectivity between disparate domains. **(Right)** When α and β (which measure the connectivity between images of the same class or same domain respectively) are larger than γ (different domain and different class), fine-tuning with labeled source data (sketch: green, filled) in the feature space achieves high accuracy on the target (real: gray, hollow).

Contrastive learning for self-supervised learning. Contrastive learning methods (Chen et al., 2020a; He et al., 2020; Caron et al., 2020) perform self-supervised learning of a feature encoder on unlabeled data. Broadly, the contrastive learning objective aims to learn an encoder that maps augmentations (e.g., crops) of the same input to similar features and augmentations of random inputs to different features. The pre-trained encoder is then fine-tuned on a downstream dataset like ImageNet (Russakovsky et al., 2015) where the downstream train and test examples come from the *same* distribution. Several theoretical works provide guarantees on the performance of contrastive learning in this setting of identical train and test distributions (Arora et al., 2019; Tosh et al., 2021; HaoChen et al., 2021). In contrast, we consider contrastive pre-training for domain adaptation, where the downstream training data are from the source domain and the test data are from the target domain.

Domain adaptation with self-supervision. Prior works have explored the application of self-supervision to UDA. However, to the best of our knowledge they have all attempted to minimize domain distance either as part of their objective (Kang et al., 2019; Wang et al., 2021; Thota & Leontidis, 2021) or for model selection (Sun et al., 2019). Conversely, contrastive pre-training has no explicit objective involving domain distance and is still competitive with conventional UDA methods, a phenomenon that is explained by our connectivity theory. Recent work (Mishra et al., 2021) has shown that in the semi-supervised domain adaptation setting (where a small numbers of target labels are given), a method based on self-supervised rotation prediction and input consistency regularization simultaneously achieves higher accuracy and learns features with larger domain distance than prior methods.

Assumptions about distribution shift. A classical assumption is covariate shift (Shimodaira, 2000), a setting in which the marginal distribution over $\mathcal{X}$ can change between domains but the label conditioned on x is fixed. However, if the source and target have disjoint support then covariate shift is trivially satisfied, and therefore recent works have proposed alternative definitions. Kumar et al. (2020) assumes the shift is gradual in a Wasserstein metric and does not explicitly assume covariate shift. Wei et al. (2021) analyzes UDA with two assumptions: 1) the source classifier has reasonable accuracy on the target, and the target satisfies an "expansion" condition. Cai et al. (2021) assume that there is significant overlap (expansion) between the source and target after augmentations and analyzes self-training for UDA. Our theoretical connectivity model is an alternative method for describing distribution shift so that contrastive pre-training adapts even under large $\mathcal{H}\Delta\mathcal{H}$-divergence.

3. Setup

3.1. Unsupervised Domain Adaptation (UDA)

We consider a classification problem from an input space $\mathcal{X} \subseteq \mathbb{R}^d$ to a label space $\mathcal{Y} = \{1,...,r\}$.

Data, model, and metrics. Each $x \in \mathcal{X}$ has a deterministic label $y_x \in \mathcal{Y}$ and domain $d_x \in \{1,2\}$ (where domain 1 is the source and domain 2 is the target). Let P_S and P_T denote the source and target input distributions respectively (both over $\mathcal{X}$). Let $P_U = \theta P_S + (1-\theta) P_T$ for some $\theta \in [0,1]$ be a mixture of the source and target domains. This mixture distribution serves as the *unlabeled* data distribution over both source and target domains.

We consider the unsupervised domain adaptation (UDA) setting: algorithms have access to labels y_x for source examples $x \sim P_S$ but not for target examples $x \sim P_T$. Algorithms are allowed to use *unlabeled* data from all domains. The goal is to learn a classifier $f : \mathcal{X} \to \mathbb{R}^r$ with low error on the target distribution $\mathcal{L}_{0-1}(f) = \mathbb{E}_{x \sim P_T}[\mathbf{1}[\text{argmax}_i f(x)_i \neq y_x]]$.

Augmentations. Modern ML models are trained with augmentations such as crops and color distortions (Krizhevsky et al., 2012; Chen et al., 2020a). For any input $x \in \mathcal{X}$, let $\mathcal{A}(\cdot \mid x)$ be the distribution of its augmentations (also over $\mathcal{X}$).

3.2. Methods

Consider a loss function $\ell : \mathbb{R}^r \times \mathcal{Y} \to \mathbb{R}$ (e.g., cross entropy or squared loss).

Empirical risk minimization (ERM) minimizes the loss ℓ over augmentations $x' \sim \mathcal{A}(\cdot \mid x)$ of source examples $x \sim P_S$ and the original labels y_x:

$$\mathcal{L}_{\text{ERM}}(f) = \mathbb{E}_{x \sim P_S, x' \sim \mathcal{A}(\cdot|x)}[\ell(f(x'), y_x)], \quad (1)$$

ERM learns a classifier $\widehat{f}_{\text{erm}} \in \text{argmin}_f \mathcal{L}_{\text{ERM}}(f)$. Augmentations improve the target accuracy of ERM and our results also hold without augmentations.

Domain adversarial neural networks (DANN) (Ganin et al., 2016) optimizes the sum of two terms: a source classification loss, and a "domain confusion" loss that makes it difficult to predict the domain d_x from the feature $\phi(x)$.

$$\begin{aligned}\mathcal{L}_{\text{DANN}}(\phi,h,h_d) = \; & \mathbb{E}_{x \sim P_S, x' \sim \mathcal{A}(\cdot|x)}[\ell(h(\phi(x')), y_x)] \quad (2)\\ & - \lambda \mathbb{E}_{x \sim P_U, x' \sim \mathcal{A}(\cdot|x)}[\ell(h_d(\phi(x')), d_x)],\end{aligned}$$

where $\lambda > 0$ is a tradeoff parameter between the two losses and h_d is a domain classifier. The classifier $\widehat{f}_{\text{dann}} = \widehat{h}_{\text{dann}} \circ \widehat{\phi}_{\text{dann}}$ is learned by minimizing the loss: $\widehat{\phi}_{\text{dann}}, \widehat{h}_{\text{dann}} \in \text{argmin}_{\phi,h} \max_{h_d} \mathcal{L}_{\text{DANN}}(\phi,h,h_d)$.

Contrastive pre-training aims to learn useful features on the unlabeled data from both source and target by training an encoder which maps data-augmented views of the same input x to similar features (the "positive pairs") and data-augmented views of random pairs of inputs to dissimilar features (the "negative pairs"). Formally, let S_+ be the distribution (on $\mathcal{X} \times \mathcal{X}$) of "positive pairs" (augmentations of a single input $\bar{x}$), given by:

$$S_+(x, x^+) = \mathbb{E}_{\bar{x} \sim P_U}[\mathcal{A}(x \mid \bar{x}) \mathcal{A}(x^+ \mid \bar{x})]. \quad (3)$$

We apply contrastive pre-training to UDA—we pretrain an encoder ϕ on source and target *unlabeled* data from P_U, and then fine-tune a classification head h on only source labeled data from P_S.

1. (Pre-train on unlabeled source and target data) We first *pre-train* an encoder $\phi : \mathcal{X} \to \mathbb{R}^k$ to minimize the distance d_+ between postive pairs, and maximize the distance d_- between random pairs of inputs

$$\begin{aligned}\mathcal{L}_{\text{pretrain}}(\phi) \triangleq \; & \mathbb{E}_{(x,x^+) \sim S_+}\left[d_+(\phi(x), \phi(x^+))\right] \quad (4)\\ & - \mathbb{E}_{x \sim P_U, x' \sim P_U}[d_-(\phi(x), \phi(x'))],\end{aligned}$$

   where the learned encoder is $\widehat{\phi} = \text{argmin}_\phi \mathcal{L}_{\text{pretrain}}(\phi)$.

2. (Fine-tune on labeled source data) We learn a classification head $\widehat{h} = \text{argmin}_h \mathcal{L}(h)$ on pretrained features $\widehat{\phi}(x)$ and their labels y_x where the loss is

$$\mathcal{L}(h) \triangleq \mathbb{E}_{x \sim P_S}[\ell(h(\widehat{\phi}(x)), y_x)]. \quad (5)$$

   The final classifier is $\widehat{h} \circ \widehat{\phi}$. In our experiments, we fine-tune both the head h and encoder ϕ.

Dataset	Feature space learned by	Across class (β)	Across domain (α)	Across both (γ)
Living-17	Input space	21.43	36.58	19.55
	DANN+strong-augs	3.44	18.00	3.38
	SwAV	2.06	12.67	1.26
DomainNet	Input space	32.88	27.36	25.12
	DANN+strong-augs	5.65	13.64	3.89
	SwAV	7.03	7.54	2.46

Table 1: Average error of classifying augmented images of two class-domain pairs (as a proxy for connectivity) in the input space and feature spaces learned by DANN+strong-augs and SwAV (contrastive pre-training). Classes are very distinguishable in both learned feature spaces, but the domains are much more difficult to distinguish in the DANN feature space than in the SwAV feature space.

4. Intuitions and Analysis

In this section, we explain when contrastive pre-training can learn transferable features—i.e., simply training a classifier on labeled source data leads to good predictions on the *unlabeled* target domain—despite keeping the source and target features very different. We extend the spectral contrastive learning framework (HaoChen et al., 2021) to incorporate distribution shift—we give intuitions in Section 4.2, a proof for the stochastic block model setting in Section 4.3, and a setting where contrastive pre-training outperforms domain adversarial neural networks (Ganin et al., 2016) (even when both use the same data augmentations), in Section 4.4.

4.1. Theoretical setup

Spectral contrastive learning. HaoChen et al. (2021) theoretically analyzed contrastive learning from the perspective of an augmentation graph in which the inputs x from $\mathcal{X}$ constitute the nodes, and the edge weight $S_+(x,x')$ represents the probability of the inputs being selected as a positive pair (augmentations of the same image x). As in HaoChen et al. (2021), we analyze the spectral contrastive learning objective, which achieves similar empirical results to other contrastive learning methods but is more amenable to theoretical analysis:

$$\begin{aligned}\mathcal{L}_{\text{pretrain}}(\phi) = -2\cdot\mathbb{E}_{(x,x^+)\sim S_+}\left[\phi(x)^\top\phi(x^+)\right]\\ +\mathbb{E}_{x,x'\sim P_U}\left[\left(\phi(x)^\top\phi(x')\right)^2\right].\end{aligned} \tag{6}$$

The pre-training step learns the encoder $\widehat{\phi}:\mathcal{X}\to\mathbb{R}^k$ by minimizing (6).

Linear classification. After pre-training, we train a linear probe on source domain features with parameters $B\in\mathbb{R}^{r\times k}$ (where k is the feature dimension and r is the number of classes). We learn the parameters $\widehat{B}$ by minimizing

$$\mathcal{L}(B)=\mathbb{E}_{x\sim P_S}\left[\ell(B\widehat{\phi}(x),y_x)\right]+\eta\|B\|_F^2, \tag{7}$$

where ℓ is the squared loss. The resulting classifier is $\widehat{f}(x)=\text{argmax}_{i\in[r]}(\widehat{B}\widehat{\phi}(x))_i$.

4.2. Simple example

We first consider a toy example that captures the core intuitions of our theory. For this example, we compute an exact (closed-form) expression for the representations learned by contrastive pre-training on unlabeled data, which are visualized in Figure 2. We then show that a linear classifier trained on source representations also gets high accuracy on examples from the target domain, *without using any target labels*.

Setup. The goal is to classify between images of clocks and butterflies—see Figure 2. The input space $\mathcal{X}$ consists of 4 points: [clock sketch, butterfly sketch, clock photo, butterfly photo], where the i-th input refers to the i-th element in the list (e.g., $i=2$ corresponds to butterfly sketch). The label space $\mathcal{Y}$ contains two classes, clock ($y_x=1$) and butterfly ($y_x=2$). We only have labeled data for the source inputs (sketches, $d_x=1$), and the goal is to achieve high accuracy on the target inputs (photos, $d_x=2$). The source distribution P_S places equal probability on sketches ($P_S(x)=0.5$ if $d_x=1$ and $P_S(x)=0$ otherwise), and the target distribution P_T places equal probability on photos ($P_T(x)=0.5$ if $d_x=2$ and $P_T(x)=0$ otherwise). The unlabeled distribution P_U places equal probability on all images (sketches and photos).

Augmentations. The key ingredient in contrastive pretraining is the augmentation strategy—contrastive pretraining aims to map augmentations of the same input x to similar representations. These augmentations include aggressive crops which keep only a small part (8%) of an image (e.g., the tail or fur of an animal), and can blur the line between classes. Augmentations such as color jitter, which also randomly 'drops' colors, can also transform the style of an image, blurring the line between domains such as sketches and photos. As such, we consider augmentations that can change the class or domain of an image. We define the probability that an image x can augment to an image x' as follows:

$$A(x'\,|\,x)=\begin{cases}\rho' & \text{if} \quad y_x=y_{x'},d_x=d_{x'}\\ \alpha' & \text{if} \quad y_x=y_{x'},d_x\neq d_{x'}\\ \beta' & \text{if} \quad y_x\neq y_{x'},d_x=d_{x'}\\ \gamma' & \text{if} \quad y_x\neq y_{x'},d_x\neq d_{x'}\end{cases},$$

Key condition for contrastive pre-training in the UDA setting. We explain in our simple example how contrastive pre-training achieves zero target error when $\rho'>\max\{\alpha',\beta'\}$ and $\min\{\alpha',\beta'\}>\gamma'$, and we prove a more general version in Section 4.3. The first condition is satisfied when an augmented image is more likely to stay within the same domain and class, than change the domain or class. The second condition is satisfied when data augmentation is less likely to change both the domain *and* class of an image than just one of domain or class. For example, an augmentation is less likely to augment a clock photo into a butterfly sketch, since it involves changing both the style (sketch vs. real) and the semantic content (butterfly vs. clock).

Deriving a closed-form expression for pre-trained representations. Consider a 4-node graph G where the nodes are input examples in $\mathcal{X}$ and where the edge weight between $x, x^+ \in \mathcal{X}$ is the probability of sampling augmentations x and x^+ from the same original image $\bar{x}$ (Equation 3): $S_+(x, x^+)$. We illustrate this graph in Figure 2 (left). Let A denote the weighted adjacency matrix for G, which depends on the augmentation parameters $\rho', \alpha', \beta', \gamma'$:

$$A = \begin{bmatrix} \rho & \beta & \alpha & \gamma \\ \beta & \rho & \gamma & \alpha \\ \alpha & \gamma & \rho & \beta \\ \gamma & \alpha & \beta & \rho \end{bmatrix}, \text{ where } \begin{cases} \rho = \frac{1}{4}\left(\rho'^2 + \alpha'^2 \right. \\ \qquad \left. + \beta'^2 + \gamma'^2\right) \\ \alpha = \frac{1}{2}\left(\rho'\alpha' + \beta'\gamma'\right) \\ \beta = \frac{1}{2}\left(\rho'\beta' + \alpha'\gamma'\right) \\ \gamma = \frac{1}{2}\left(\rho'\gamma' + \alpha'\beta'\right) \end{cases}$$

Recall that we assumed $\rho' > \max\{\alpha', \beta'\}$ and $\min\{\alpha', \beta'\} > \gamma'$. With some algebra, this implies that the same condition holds for the positive pair probabilities as well: $\rho > \max\{\alpha, \beta\}$ and $\min\{\alpha, \beta\} > \gamma$.

Contrastive pre-training learns an encoder $\widehat{\phi} : \mathcal{X} \to \mathbb{R}^k$ given by Equation 6—we use $k = 3$ here. Let $\widehat{\Phi} \in \mathbb{R}^{4 \times k}$ be the learned feature matrix, where the i-th row of $\widehat{\Phi}$ contains representations for example i, so $\widehat{\Phi}_i = \widehat{\phi}(i)$.

From a few lines of algebra (Lemma 3.2 in HaoChen et al. (2021)), $\mathcal{L}_{\text{pretrain}}(\widehat{\phi}) = \frac{1}{16}\|16A - \widehat{\Phi}\widehat{\Phi}^\top\|_2^2 + c$, where c is a constant that does not depend on $\widehat{\phi}$. Here, A has rank 4, while $\widehat{\Phi}\widehat{\Phi}^\top$ has rank 3—the minimizer $\widehat{\Phi}$ is given by the eigenvectors corresponding to the 3 largest eigenvalues of A. We can compute the (unordered) eigenvalues $\lambda_a, \lambda_b, \lambda_c, \lambda_d$ and corresponding eigenvectors u_a, u_b, u_c, u_d of A explicitly:

$$\begin{aligned} &\lambda_a = \alpha + \beta + \gamma + \rho, && \lambda_b = -\alpha + \beta - \gamma + \rho, \\ &\lambda_c = \alpha - \beta - \gamma + \rho, && \lambda_d = -\alpha - \beta + \gamma + \rho, \\ &u_a = (1,1,1,1)^\top, && u_b = (1,1,-1,-1)^\top, \\ &u_c = (1,-1,1,-1)^\top, && u_d = (1,-1,-1,1)^\top. \end{aligned}$$

Since $\beta > \gamma$ and $\alpha > \gamma$, we have:

1. $\lambda_a > \lambda_d$ since all $\rho, \beta, \alpha, \gamma$ are nonnegative,
2. $\lambda_b > \lambda_d$ when $\beta - \gamma > -\beta + \gamma$, which is implied by $\beta > \gamma$,
3. $\lambda_c > \lambda_d$ when $\alpha - \gamma > -\alpha + \gamma$, which is implied by $\alpha > \gamma$.

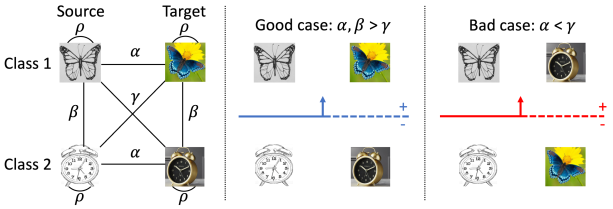

Figure 2: **(Left)** Illustrative example in binary classification, where each class-domain pair is a single node in the graph. Edge weights denote connectivity (probability of sampling the endpoints as a positive pair). The self-loop weight ρ denotes the probability of sampling a pair of the same domain and class. **(Middle)** When α (same class, different domain) and β (different class, same domain) are greater than γ (different domain, different class), the features are oriented so that a source-trained linear classifier generalizes to the target. The class and domain information are disentangled along the vertical and horizontal axes, respectively. **(Right)** When $\alpha < \gamma$, the target features are flipped and the source-trained classifier does not generalize.

So λ_d is the smallest eigenvalue, and the learned features are given by (up to rotational symmetries):

$$\widehat{\Phi} = \begin{bmatrix} \widehat{\phi}(\text{clock sketch})^\top \\ \widehat{\phi}(\text{butterfly sketch})^\top \\ \widehat{\phi}(\text{clock photo})^\top \\ \widehat{\phi}(\text{butterfly photo})^\top \end{bmatrix} = 4 \begin{bmatrix} \sqrt{\lambda_a} & \sqrt{\lambda_b} & \sqrt{\lambda_c} \\ \sqrt{\lambda_a} & \sqrt{\lambda_b} & -\sqrt{\lambda_c} \\ \sqrt{\lambda_a} & -\sqrt{\lambda_b} & \sqrt{\lambda_c} \\ \sqrt{\lambda_a} & -\sqrt{\lambda_b} & -\sqrt{\lambda_c} \end{bmatrix}$$

Fine-tuning on source examples gets zero error on target. So far, we have calculated the features learned by contrastive pretraining. We note that the first feature dimension is the same for all examples and therefore not useful—we visualize the remaining 2-D features in the middle panel of Figure 2. We see that the features disentangle the domain and class information—the second feature dimension can be used to distinguish domains while the third feature can distinguish classes. Therefore, training a max-margin classifier $\widehat{f}$ on only labeled sketch images (first two rows of the feature matrix $\widehat{F}$) produces a classifier that uses only the third feature, which extrapolates to photos correctly and gets *zero OOD error* ($\mathcal{L}_{0-1}(\widehat{f}) = 0$). The key difference from conventional intuitions in domain adaptation is that contrastive pre-training does not learn domain-invariant features—the source and target are perfectly distinguishable but the domain and class information are disentangled, enabling generalization.

Note that if our key condition on the augmentations does not hold e.g., $\alpha < \gamma$ but $\gamma < \beta$, then the learned features $\widehat{F}$ exclude the eigenvector corresponding to λ_c as the smallest. Again, the first feature is constant and not useful. In the visualization (Figure 2 right), we show the features learned by contrastive pre-training. In this case, a max-margin classifier $\widehat{f}$ trained on sketch images (red line) mislabels the photos ($\mathcal{L}_{0-1}(\widehat{f})=1$) because the orientation of the classes is flipped between domains.

4.3. Theory for multi-class stochastic block model

We extend the simple example in Section 4.2 to a stochastic block model setting with r classes, where we have n points for each class and domain. The unlabeled data distribution P_U is the uniform distribution over $\mathcal{X}$.

The stochastic block model (SBM) (Holland et al., 1983) is a random graph model widely used in theoretical computer science to model communities, networks, and topic models in NLP (Bouveyron et al., 2016; Abbe, 2018; Mehta et al., 2019). The graph $G=(\mathcal{X},E)$ is defined as follows: for each pair of nodes $x,x'\in\mathcal{X}$, the corresponding edge (undirected, unweighted) is in the edge set E with probability:

$$\begin{cases} \rho & \text{if} \quad y_x=y_{x'}, d_x=d_{x'} \\ \alpha & \text{if} \quad y_x=y_{x'}, d_x\neq d_{x'} \\ \beta & \text{if} \quad y_x\neq y_{x'}, d_x=d_{x'} \\ \gamma & \text{if} \quad y_x\neq y_{x'}, d_x\neq d_{x'} \end{cases},$$

and all edges are sampled independently. The distribution of positive pairs S_+ is the uniform distribution over E.

Our key condition is that 1) augmentations are less likely to change both the domain and class ($\min\{\alpha,\beta\}>\gamma$) and 2) augmentations are more likely to keep both class and domain unchanged than to change either one of the two ($\rho>\max\{\alpha,\beta\}$) . Assuming this and using a feature dimension $r+1$, the following result shows that contrastive pre-training on unlabeled source and target data learns transferable features so that a linear head trained on the source features achieves high accuracy on the target.

Theorem 1. *In the SBM setting above, let $\rho>\max\{\alpha,\beta\}$ and $\min\{\alpha,\beta\}>\gamma$ and let the pre-training feature dimension be $r+1$. Then, for any $n\geq\Omega\left(\frac{r}{\min\{\alpha-\gamma,\beta-\gamma\}^2}\right)$ and regularization strength $\eta\in\left(0,\frac{\alpha-\gamma}{8r\rho}\right)$, with probability at least 0.999, we have*

$$\mathcal{L}_{0-1}(\widehat{f})\leq O\left(\frac{1}{\eta^4\cdot(\min\{\alpha,\beta\}-\gamma)^2\cdot n}\right)\cdot poly(r).$$

where $\widehat{f}(x)$ is the linear classifier trained on source domain pre-trained features (Section 4.1).

The rate at which the error decreases with the number of samples n (per class-domain pair) is controlled by the gap between the across-class/across-domain connectivities β,α and the across-both connectivity γ. Thus, augmentations that tend to change only one of class or domain will improve the downstream transferability of the learned features.

Proof sketch. The features learned by contrastive pre-training are given by the top k eigenvectors of the adjacency matrix A, which is a random matrix defined by the SBM. For the expected adjacency matrix $\mathbb{E}[A]$, similarly to the simple example, we can compute the eigenvectors in closed form and show that the linear head learned on the source data achieves low error on the target. The main challenge is to show an analogous result for the *sampled* graph, where each of the data points per class-domain pair can have a different set of edges. We use matrix concentration bounds to concentrate the top eigenvectors of A to those of $\mathbb{E}[A]$ and use matrix perturbation analysis to show that the predictor learned using A is close to the "ideal" one learned using $\mathbb{E}[A]$. This shows that contrastive pre-training on the random graph defined by A also learns transferable features with high probability, which gives the result. □

The full proof is in Appendix B.1, where we show that the result holds even when we have more than 2 domains. This suggests that we can pre-train one model on the unlabeled data of many domains and the features can transfer across all of them. Our bound in the appendix also holds with probability arbitrarily close to 1.

As discussed earlier, contrastive pre-training does not merge source and target features. Our next result mirrors this observation theoretically in the SBM setting, showing that a linear classifier on top of the pre-trained features can classify the domain with low error. Letting $h_d:\mathbb{R}^k\rightarrow\{1,2\}$ be a domain classifier , we define the 0-1 loss for domain classification as $\mathcal{L}^D_{0-1}(h_d\circ\widehat{\phi})=\mathbb{E}_{x\sim P_U}[\mathbf{1}[h_d(\widehat{\phi}(x))\neq d_x]]$.

Proposition 2. *In the setting of Theorem 1, with probability at least 0.999, there exists a linear classifier $h_d:\mathbb{R}^k\rightarrow\{1,2\}$, such that h_d composed with the encoder $\widehat{\phi}$ can distinguish the domains:*

$$\mathcal{L}^D_{0-1}(h_d\circ\widehat{\phi})\leq O\left(\frac{1}{\min\{\alpha-\gamma,\beta-\gamma\}^6\cdot n}\right)\cdot poly(r).$$

The proof is in Appendix B.1 and is similar to Theorem 1.

4.4. Setting where ERM/DANN underperform contrastive pre-training

We theoretically show a simple setting in which contrastive learning on unlabeled data can have higher target extrapolation accuracy than both ERM and DANN (Equations 1 and 2, respectively). The main intuition of our construction is that ERM and DANN may underperform when there are subsets of the target data distribution that are unreachable via data augmentation on any labeled source input—in other words, inputs x with $d_x=2$ such that $\mathcal{A}(x\mid x')=0$ for all x' with $d_{x'}=1$. This makes the problem underconstrained for both ERM and DANN, while contrastive pre-training better leverages the connectivity information from augmentations on unlabeled source and target inputs to learn transferable features.

Proposition 3. *There exists a set $\mathcal{X}$, a distribution P_U and data augmentation $\mathcal{A}$, such that for some feature dimension $k \in \mathbb{Z}^+$, a linear classifier trained on contrastive pre-trained features achieves 0 target error: $\mathcal{L}_{0-1}(\widehat{f}) = 0$. However, for all $k \in \mathbb{Z}^+$, there exist minimizers $\widehat{f}_{erm}$ and $\widehat{f}_{dann}$ of the ERM and DANN objectives, respectively, that have non-zero error: $\mathcal{L}_{0-1}(\widehat{f}_{erm}) = \mathcal{L}_{0-1}(\widehat{f}_{dann}) = 1/3$.*

The proof is in Appendix B.2. For ERM, an optimal predictor that minimizes the ERM objective (1) can make arbitrary predictions on unseen target examples. For DANN, simply matching the marginal input distributions (even with augmentations) across domains can potentially swap the classes of the unreachable target data—this is consistent with the well-known drawback that DANN with an expressive feature extractor can learn domain-invariant features by arbitrarily pairing the source and target examples (Shu et al., 2018; Zhao et al., 2019). Conversely, contrastive pre-training learns features that enable perfect transferability.

5. Contrastive pre-training is a strong domain adaptation method

Our theory gives conditions for which contrastive learning produces effective features for domain adaptation. We now empirically validate contrastive pre-training as a competitive domain adaptation method. We observe that contrastive pre-training achieves comparable performance to strong UDA methods on four standard domain adaptation datasets.

Datasets. We conduct experiments on DomainNet (Peng et al., 2019; Prabhu et al., 2021), which contains 40 classes and 4 domains, BREEDS Living-17 and Entity-30 (Santurkar et al., 2020), which are adaptation benchmarks derived from ImageNet, and STL-10→CIFAR-10 (Coates et al., 2011; Krizhevsky, 2009; French et al., 2018), which are two classical image recognition datasets often paired together for domain adaptation.

Contrastive pre-training algorithm. We use SwAV (Caron et al., 2020), a contrastive pre-training algorithm with high accuracy on ImageNet, for our ImageNet-like datasets (BREEDS and DomainNet). SwAV uses a multi-crop data augmentation strategy with several crops of different sizes, followed by horizontal flipping, color distortion, and Gaussian blurring. We pre-train on the combined source and target unlabeled data and fine-tune on the source using the same augmentation pipeline used during pre-training. For STL→CIFAR, we use a publicly available SimCLR (Chen et al., 2020a) model that is pre-trained on CIFAR. SimCLR applies random cropping, color distortions, and Gaussian blur for data augmentation. In our theory we linear probe with the squared loss, but in our experiments we follow a more standard approach and fine-tune the model (pre-trained encoder and randomly initialized linear head) with the cross-entropy loss.

Baselines. We compare with standard ERM on the labeled source data and two strong domain adaptation methods: DANN (Ganin et al., 2016) and SENTRY (Prabhu et al., 2021). SENTRY achieves SoTA results on DomainNet when initialized with ImageNet-pre-trained models. However, to ensure the methods all use the same data, we initialize the domain adaptation methods with the ERM baseline (no ImageNet pre-training). For STL→CIFAR, we compare to Dirt-T (Shu et al., 2018), a SoTA domain adaptation algorithm for this task. For a more complete comparison, we consider not only using each method's default augmentations but also using the contrastive pre-training augmentations to all baselines (denoted +strong-augs).

5.1. Results

Main comparison (Table 2). On average over all pairs of DomainNet, SwAV is within 1% of the best baseline, DANN+strong-augs (44.91% vs. 45.81%). Contrastive pre-training is comparable with the best baseline (SENTRY) on Living-17 (75.12% vs. 75.53%) and improves over the best baseline (SENTRY+strong-augs) by 3.1% on Entity-30. On STL→CIFAR, the SimCLR target accuracy (75.4%) is very close to the Dirt-T accuracy reported in the original paper (75.3%). We find that replacing the default data augmentation used by each baseline with the augmentations used by contrastive pre-training consistently boosts performance for DANN but only sometimes boosts performance for SENTRY. Table 4 provides the individual results for each pair of domains on DomainNet, and Tables 4 and 9 contain results from additional contrastive pre-training methods, MoCo-V2 (Chen et al., 2020b) and MoCo-V3 (Chen et al., 2021) (all trained on the same data).

Extra unlabeled data from related domains. We also consider adding extra unlabeled data from other (related) domains in DomainNet, Living-17, and Entity-30. In particular, we pre-train once on an unlabeled dataset consisting of a superset of the domains we want to adapt to, fine-tune on labeled data from one of the domains, and evaluate on all other domains. While this is not a fair comparison to other domain adaptation methods, the ability to scale to large unlabeled datasets by pre-training on all domains simultaneously is a natural advantage of pre-training. This method, which we denote as SwAV+extra, gives further improvements over SwAV (nearly 7% on DomainNet, 7% on Living-17, and 4% on Entity-30). In Table 9 we show results from pre-training on different splits of the BREEDS data, as well as using additional pre-training methods: Dino+extra (Caron et al., 2021) and Barlow Twins+extra (Zbontar et al., 2021).

Method	ERM		SENTRY		DANN		Pre-training	SwAV+extra
Augmentations	Standard	Strong	Standard	Strong	Standard	Strong	Strong	Strong
DomainNet (avg. of 12 pairs)	26.67	37.68	31.58	25.82	38.38	**45.81**	44.91	51.73
Living-17	60.17	63.29	**75.53**	74.53	65.59	71.29	75.12	82.00
Entity-30	52.68	52.52	56.10	58.90	57.45	57.52	**62.03**	65.90
STL-10→CIFAR-10	44.83	57.40	53.84	43.71	46.77	55.18	**75.41**	N/A

Table 2: Test accuracy (%) of baselines and contrastive pre-training on 4 benchmark visual adaptation datasets. The first and second highest numbers for each dataset are bolded and underlined, respectively. The pre-training algorithm is SwAV for DomainNet, Living-17, and Entity-30, and SimCLR for STL→CIFAR. Contrastive pre-training is competitive with the baselines for all datasets. On STL→CIFAR, SimCLR is slightly higher than Dirt-T, the previous SoTA (75.3% as reported in Shu et al. (2018)). SwAV+extra is not bolded as it uses additional data and is therefore not directly comparable.

6. Evaluating connectivity on real datasets

Recall that our theory (Theorem 1) provides conditions on connectivities between classes and domains for which contrastive pre-training provably obtains good target accuracy. We heuristically estimate each of the connectivity measures on Living-17 and DomainNet to verify the predictions from our theory in Section 4. We show that the empirical connectivities satisfy our theoretical conditions for contrastive pre-training to learn transferable features (e.g., across-domain > across-both connectivity) and that the connectivity ratios (e.g., across-domain / across-both) are predictive of target domain accuracy of contrastive pre-training.

Estimating connectivity on benchmark datasets. To verify whether real datasets and augmentations satisfiy the connectivity requirements for Theorem 1, we compute empirical estimates of the connectivity measures. Using augmented images from 2 class-domain pairs, we train a classifier to predict the class-domain pair from which the image originated. For example, to estimate across-domain connectivity (α from Section 4.2), we choose 2 class-domain pairs with the same class and different domain. The estimates are in Table 1 (see Appendix E for full details). We find that connectivity ratios satisfy our theoretical conditions in all cases.

Connectivity ratios correlate with target accuracy. Section 4 suggests that when the across-domain (α) and across-class (β) connectivities are larger than the across-both (γ) connectivity (i.e., α/γ and β/γ are large), contrastive pre-training learns features with good target accuracy. To investigate the relation between the connectivity ratios and target accuracy, we consider fitting the following function, with parameters w_1 and w_2, to the SwAV target accuracy on the 12 pairs of DomainNet domains:

$$\text{target accuracy} \approx (\alpha/\gamma)^{w_1} \cdot (\beta/\gamma)^{w_2}. \tag{8}$$

Since our theory predicts that target accuracy is high when both ratios are large, we multiply them to express the logical "and". To normalize by the differing intrinsic difficulties of transferring to each target domain of DomainNet, we fit Eq. 8 to the improvement over the worst pair over all source/target pairs with the same target domain. Linear regression in log space yields $w_1 = 14.9$ and $w_2 = 2.7$, with a strong coefficient of determination ($R^2 = 0.78$) between the predicted and observed accuracies (left panel of Figure 3). Thus, the target accuracy can be well-explained *using the connectivity ratios alone*.

Figures 5 and 6 show that the predicted target accuracies using this method are also accurate for MoCo-V2 (Chen et al., 2020b) ($R^2 = 0.79$) and MoCo-V3 (Chen et al., 2021) ($R^2 = 0.60$), but much less accurate for the baselines (both with and without augmentations; average $R^2 = 0.21$).

Ablating connectivity degrades target accuracy. We find that the across-domain connectivity is very important, as those examples "bridge" the domains—for instance, our linear regression fit puts the largest weight on the across-domain connectivity ratio. We verify this intuition by training a domain classifier on Living-17 and pre-training on the subset of examples on which the classifier is most confident (i.e., the examples farthest from the classifier's decision boundary, which intuitively contribute the least to across-domain connectivity). As controls, we also pre-train on 1) random subsets of the same size and 2) the examples on which the classifier is least confident. For 5 different subset sizes, our confidence pruning method consistently reduces target accuracy compared to both controls (Figure 3 right).

Pre-trained features approximately disentangle class and domain. In Section 4, we showed theoretically that contrastive pre-training can learn a feature space that is simultaneously predictive of the class and domain by disentangling the information along separate directions. Here, we verify this empirically using the SwAV feature space. Given a source and target dataset, we test disentanglement by training the following linear classifiers on the feature space: 1) Source classifier, which predicts class in the source domain; 2) Target classifier, which predicts class in the target

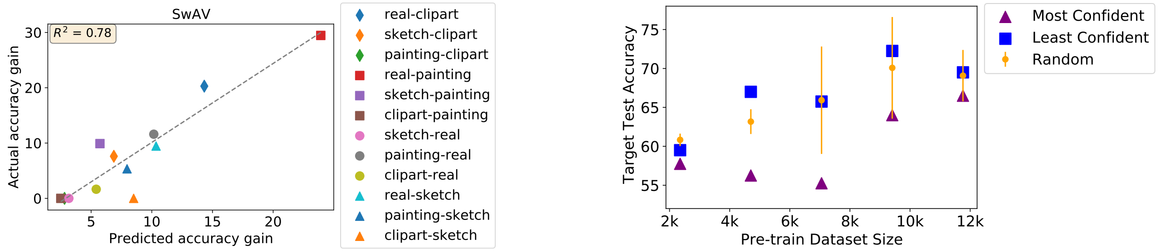

Figure 3: **(Left)** True target accuracy of SwAV (gain over the worst source/target pair with the same target) vs. the predicted accuracy using the connectivity ratios on 12 pairs of DomainNet domains. The predictions have a high coefficient of determination with the observed ($R^2 = 0.78$). **(Right)** Ablation of connectivity: pre-training only on inputs far from the margin of a domain classifier ("most confident") consistently degrades target accuracy more than both on random subsets (mean and standard deviation over 3 random subsets shown) and on inputs nearest the margin ("least confident").

	Class (S) vs. Class (T)	Class (S) vs. Domain	Class (T) vs. Domain
Living-17	0.397	0.013	0.016
DomainNet (avg.)	0.187	0.018	0.018

Table 3: Cosine similarity of class and domain classifiers trained on SwAV representations on Living-17 and DomainNet (average over all classes). Linear classifiers trained to predict the class on the source and target representations individually learn similar weights and are nearly orthogonal to the linear weights learned by domain classifiers.

domain;[1] and 3) Domain classifier, which predicts domain (source or target). If the class and domain information are approximately disentangled along different dimensions, then the linear weights of the source/target classifier should be orthogonal to the weights of the domain classifier.

We train the linear classifiers and compute the cosine similarity between their weights on Living-17 and all pairs of domains from DomainNet (Table 3). First, we find that the domain classifier is nearly orthogonal to the source and target classifiers, with an average cosine similarity < 0.02 between the linear weights of the domain classifier and the weights of the source and target classifiers. Second, we find that the source and target classifiers are relatively well-aligned: on average over all classes, the cosine similarity of linear weights for the same class from the source and target classifiers is high (0.40 on Living-17 and average 0.19 on DomainNet). Full results are in Appendix D.

7. Conclusion

While off-the-shelf contrastive pre-training is not intentionally a domain adaptation method, we find that not only does it theoretically and empirically transfer well across distribution shifts, but it does so in a way that runs counter to conventional domain adaptation theory and practice by not learning domain-invariant features. Given some of the practical advantages of pre-training (such as being able to pre-train once and then finetune for many different downstream tasks), we hope that our connectivity theory leads to improvements for contrastive pre-training as a domain adaptation method, such as improving pre-training data selection, developing augmentations to increase connectivity, and improving fine-tuning methods that exploit the geometry of the pre-trained feature space. Our connectivity theory can also potentially explain when contrastive pre-training for domain adaptation does *not* perform well, such as on the iWildCam2020-WILDS dataset (Sagawa et al., 2022). Future work may also lead to new domain adaptation methods that focus on increasing connectivity rather than collapsing domains.

Our theoretical setup is highly stylized and the stochastic block model may not be a realistic assumption; in particular, it requires a uniform marginal input distribution. Follow-up work (HaoChen et al., 2022) proves the linear transferability of contrastive pre-training in a more general setting and only requires that the same-class across-domain connectivity is larger than the across-class across-domain connectivity (rather than also requiring across-class same-domain connectivity to be large). Their analysis leads to an improved linear probing algorithm which outperforms both SwAV and MoCo-V3 on Entity-30.

8. Acknowledgements

This work was in part supported by the Open Philanthropy Project and NSF Award Grant No. 1805310, and NSF IIS 2045685. KS was also supported by the Stanford Computer Science department's CURIS program. AK was also supported by the Rambus Corporation Stanford Graduate Fellowship. SMX was also supported by an NDSEG Fellowship. We would like to thank Kaylee Burns, Aditi Raghunathan, and the anonymous reviewers, for helpful comments on our draft.

[1] Although this cannot be done in practice, we only use the target labels here for exploratory analysis.

References

Abbe, E. Community detection and stochastic block models: Recent developments. *Journal of Machine Learning Research*, 18(177):1–86, 2018. URL http://jmlr.org/papers/v18/16-480.html.

AlBadawy, E., Saha, A., and Mazurowski, M. Deep learning for segmentation of brain tumors: Impact of cross-institutional training and testing. *Med Phys.*, 45, 2018.

Arora, S., Khandeparkar, H., Khodak, M., Plevrakis, O., and Saunshi, N. A theoretical analysis of contrastive unsupervised representation learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 5628–5637, 2019.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., and Vaughan, J. W. A theory of learning from different domains. *Machine Learning*, 79(1):151–175, 2010.

Bouveyron, C., Latouche, P., and Zreik, R. The stochastic topic block model for the clustering of vertices in networks with textual edges. *Statistics and Computing*, 2016. doi: 10.1007/s11222-016-9713-7. URL https://hal.archives-ouvertes.fr/hal-01299161.

Cai, T., Gao, R., Lee, J., and Lei, Q. A theory of label propagation for subpopulation shift. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 1170–1182. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/cai21b.html.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 9912–9924, 2020.

Caron, M., Touvron, H., Misra, I., Jegou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. In *International Conference on Computer Vision (ICCV)*, 2021.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, pp. 1597–1607, 2020a.

Chen, X., Fan, H., Girshick, R. B., and He, K. Improved baselines with momentum contrastive learning. *arXiv*, 2020b.

Chen, X., Xie, S., and He, K. An empirical study of training self-supervised vision transformers. *arXiv preprint arXiv:2104.02057*, 2021.

Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15, pp. 215–223, 2011.

Dai, D. and Gool, L. V. Dark model adaptation: Semantic image segmentation from daytime to nighttime. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 3819–3824, 2018. doi: 10.1109/ITSC.2018.8569387.

Davis, C. and Kahan, W. M. The rotation of eigenvectors by a perturbation. iii. *SIAM Journal on Numerical Analysis*, 7(1):1–46, 1970.

Engstrom, L., Ilyas, A., Salman, H., Santurkar, S., and Tsipras, D. Robustness (python library). https://github.com/MadryLab/robustness, 2019.

French, G., Mackiewicz, M., and Fisher, M. Self-ensembling for visual domain adaptation. In *International Conference on Learning Representations*, 2018.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., March, M., and Lempitsky, V. Domain-adversarial training of neural networks. *Journal of Machine Learning Research (JMLR)*, 17, 2016.

HaoChen, J. Z., Wei, C., Gaidon, A., and Ma, T. Provable guarantees for self-supervised deep learning with spectral contrastive loss. *arXiv preprint arXiv:2106.04156*, 2021.

HaoChen, J. Z., Wei, C., Kumar, A., and Ma, T. Beyond separability: Analyzing the linear transferability of contrastive representations to related subpopulations. *arXiv*, 2022.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Computer Vision and Pattern Recognition (CVPR)*, 2020.

Holland, P. W., Laskey, K. B., and Leinhardt, S. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

Kang, G., Jiang, L., Yang, Y., and Hauptmann, A. G. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4893–4902, 2019.

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1097–1105, 2012.

Kumar, A., Ma, T., and Liang, P. Understanding self-training for gradual domain adaptation. In *International Conference on Machine Learning (ICML)*, 2020.

Lei, J. and Rinaldo, A. Consistency of spectral clustering in stochastic block models. *The Annals of Statistics*, 43: 215–237, 2015.

Li, J., Chen, E., Ding, Z., Zhu, L., Lu, K., and Shen, H. T. Maximum density divergence for domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):3918–3930, 2021.

Li, M., Zhai, Y.-M., Luo, Y.-W., Ge, P.-F., and Ren, C.-X. *Enhanced transport distance for unsupervised domain adaptation*. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

Mehta, N., Duke, L. C., and Rai, P. Stochastic blockmodels meet graph neural networks. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4466–4474. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/mehta19a.html.

Mishra, S., Saenko, K., and Saligrama, V. Surprisingly simple semi-supervised domain adaptation with pretraining and consistency. *arXiv preprint arXiv:2101.12727*, 2021.

Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., and Wang, B. Moment matching for multi-source domain adaptation. In *International Conference on Computer Vision (ICCV)*, 2019.

Prabhu, V., Khare, S., Karthik, D., and Hoffman, J. Selective entropy optimization via committee consistency for unsupervised domain adaptation. In *International Conference on Computer Vision (ICCV)*, 2021.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115 (3):211–252, 2015.

Sagawa, S., Koh, P. W., Lee, T., Gao, I., Xie, S. M., Shen, K., Kumar, A., Hu, W., Yasunaga, M., Marklund, H., Beery, S., David, E., Stavness, I., Guo, W., Leskovec, J., Saenko, K., Hashimoto, T. B., Levine, S., Finn, C., and Liang, P. Extending the WILDS benchmark for unsupervised adaptation. In *International Conference on Learning Representations (ICLR)*, 2022.

Santurkar, S., Tsipras, D., and Madry, A. Breeds: Benchmarks for subpopulation shift. *arXiv*, 2020.

Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90:227–244, 2000.

Shu, R., Bui, H. H., Narui, H., and Ermon, S. A DIRT-T approach to unsupervised domain adaptation. In *International Conference on Learning Representations (ICLR)*, 2018.

Stewart, G. W. On the perturbation of pseudo-inverses, projections and linear least squares problems. *SIAM review*, 19(4):634–662, 1977.

Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., Vasudevan, V., Han, W., Ngiam, J., Zhao, H., Timofeev, A., Ettinger, S., Krivokon, M., Gao, A., Joshi, A., Zhao, S., Cheng, S., Zhang, Y., Shlens, J., Chen, Z., and Anguelov, D. Scalability in perception for autonomous driving: Waymo open dataset, 2020.

Sun, Y., Tzeng, E., Darrell, T., and Efros, A. A. Unsupervised domain adaptation through self-supervision. *arXiv*, 2019.

Tan, S., Peng, X., and Saenko, K. Class-imbalanced domain adaptation: An empirical odyssey. *arXiv preprint arXiv:1910.10320*, 2020.

Thota, M. and Leontidis, G. Contrastive domain adaptation. *arXiv*, 2021.

Tosh, C., Krishnamurthy, A., and Hsu, D. Contrastive estimation reveals topic posterior information to linear models. *Journal of Machine Learning Research (JMLR)*, 22(281):1–31, 2021.

Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., and Darrell, T. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

Tzeng, E., Hoffman, J., Saenko, K., and Darrell, T. Adversarial discriminative domain adaptation. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

Wang, R., Wu, Z., Weng, Z., Chen, J., Qi, G.-J., and Jiang, Y.-G. Cross-domain contrastive learning for unsupervised domain adaptation. *arXiv*, 2021.

Wei, C., Shen, K., Chen, Y., and Ma, T. Theoretical analysis of self-training with deep networks on unlabeled data. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=rC8sJ4i6kaH.

Xiao, N. and Zhang, L. *Dynamic weighted learning for unsupervised domain adaptation*. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021.

Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., and Darrell, T. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning (ICML)*, volume 139, 2021.

Zhao, H., des Combes, R. T., Zhang, K., and Gordon, G. J. On learning invariant representation for domain adaptation. In *International Conference on Machine Learning (ICML)*, 2019.

A. Ethics

Our experiments use standard benchmark vision datasets in domain adaptation, which is publicly available data. In general, however, large-scale unsupervised learning can involve data scraped from the internet which may lead to privacy concerns. Care should be taken when curating datasets for large scale contrastive pre-training in practice.

B. Additional details for Section 4

B.1. Proof of Theorem 1

We first summarize the setting of the multi-domain multi-class classification problem that we will analyze.

Consider a multi-way classification problem, where r is the number of classes, m is the number of domains, n is the number of data in each class of each domain. The total number of data is $N=rmn$. The set of all nodes is $\mathcal{X}$.

For data $x\in\mathcal{X}$, we use $d_x\in[m]$ and $y_x\in[r]$ to denote its domain and class, respectively.

We consider a stochastic block model for the graph, where the probability of existence of an edge between x and x' is: (1) ρ if $d_x=d_{x'}$ and $y_x=y_{x'}$, (2) α if $d_x\neq d_{x'}$ and $y_x=y_{x'}$, (3) β if $d_x=d_{x'}$ and $y_x\neq y_{x'}$, (4) γ if $d_x\neq d_{x'}$ and $y_x\neq y_{x'}$.

Let $A\in\mathbb{R}^{N\times N}$ be the adjacency matrix of the graph. A random positive pair (x,x^+) is a uniform random edge (i.e., $p_+(x,x^+)=A_{xx^+}/\sum_{x',x''}A_{x'x''}$), a negative pair (x,x^-) are two uniform random nodes. Let $k=m+r-1$ be the feature dimension and $f:\mathcal{X}\to\mathbb{R}^k$ be the feature map learned by minimizing the population spectral contrastive loss:

$$-2\mathbb{E}_{x,x^+}[f(x)^\top f(x^+)]+\mathbb{E}_{x,x^-}\left[\left(f(x)^\top f(x^-)\right)^2\right]. \tag{9}$$

Let $\mathcal{S}=\{x\in\mathcal{X}:d_x=1\}$ and $\mathcal{T}=\{x\in\mathcal{X}:d_x\neq 1\}$ be source and target domains, respectively. Given the labeled source domain data, we learn the linear head

$$\hat{h}=\operatorname*{argmin}_{h\in\mathbb{R}^{(k\times r)}}\sum_{x\in\mathcal{S}}\left(\left\|b^\top f(x)-\vec{y}_x\right\|_2^2+\eta\|b\|_F^2\right), \tag{10}$$

where $\vec{y}_x=e_{y_x}-\frac{1}{r}\cdot\mathbf{1}\in\mathbb{R}^r$ is the mean-zero one-hot embedding of the label, $\eta>0$ is the regularization strength. For data $x\in\mathcal{T}$, we use $\text{pred}(x)=\text{argmax}_i(\hat{b}^\top f(x))_i$ as the predictor.

We have the following more generalized version of Theorem 1.

Theorem 4. *Let $\zeta>0$ and $\epsilon\in\left(0,\frac{1}{2}\right)$ be arbitrary constants. In the above stochastic block model, assume $\rho>\max\{\alpha,\beta\}$ and $\gamma<\min\{\alpha,\beta\}$. Then, there exists $\tilde{\xi}\in[1-\epsilon,1]$, such that for any $n\geq\Omega\left(\frac{rm}{\min\{\alpha-\gamma,\beta-\gamma\}^2}\right)$ and regularization strength $\eta\in\left(0,\frac{(\alpha-\gamma)\epsilon}{2r\rho}\right]$, with probability at least $1-n^{-\zeta}$, we have*

$$\sum_{x\in\mathcal{T}}\left\|\hat{b}^\top f(x)-\tilde{\xi}\vec{y}_x\right\|_2^2\leq O\left(\frac{1}{\eta^4\cdot\min\{\alpha-\gamma,\beta-\gamma\}^2}\right)\cdot\text{poly}(r,m). \tag{11}$$

Furthermore, the target error can be bounded by

$$\mathbb{P}_{x\sim\mathcal{T}}(\text{pred}(x)\neq y_x)\leq O\left(\frac{1}{\eta^4\cdot\min\{\alpha-\gamma,\beta-\gamma\}^2\cdot n}\right)\cdot\text{poly}(r,m). \tag{12}$$

Note that setting $\epsilon=\frac{1}{4}$ and $\zeta=1$ proves Theorem 1. In the rest of this subsection, we will give a proof of Theorem 4.

Let $A_k\in\mathbb{R}^{N\times N}$ be the rank-k approximation of the adjacency matrix A, which contains the top k components of A's SVD decomposition. We use $A_{k,(\mathcal{T},\mathcal{S})}$ to denote the matrix by restricting A_k to the rows corresponding to the source and the columns corresponding to the target. We use $A_{k,(\mathcal{S},\mathcal{S})}$ to denote the matrix by restricting A_k to the rows and columns corresponding to the source.

Let $Y \in \mathbb{R}^{N\times r}$ be the label matrix where on the x-th row it contains the label target $\vec{y}_{x=}e_{y_x} - \frac{1}{r}\cdot\mathbf{1}$. Let $Y_\mathcal{S} \in \mathbb{R}^{|\mathcal{S}|\times r}$ and $Y_\mathcal{T} \in \mathbb{R}^{|\mathcal{T}|\times r}$ be the matrices by restricting Y to the source and target domain, respectively.

Let $\text{pred} \in \mathbb{R}^{N\times r}$ be the matrix with $\hat{b}^\top f(x)$ on its x-th row. Let $\text{pred}_\mathcal{T}$ be the matrix by restricting pred to the target domain. The following lemma gives a closed-form expression for the prediction on the target domain.

Lemma 1. *In the setting of Theorem 4, let $|E| \triangleq \sum_{x,x'} A_{xx'}$ be the total number of edges. Then,*

$$\text{pred}_\mathcal{T} = A_{k,(\mathcal{T},\mathcal{S})}\left(A_{k,(\mathcal{S},\mathcal{S})} + \frac{|E|}{N^2}\cdot\eta\cdot|S|\cdot I\right)^\dagger Y_\mathcal{S}, \tag{13}$$

where $(\cdot)^\dagger$ is the Moore–Penrose inverse, $|\mathcal{S}|$ is the number of data in the source domain.

Proof of Lemma 1. By the definition of spectral contrastive loss, we can rewrite the loss function as

$$-2\sum_{x,x'}\frac{A_{xx'}}{|E|}f(x)^\top f(x') + \sum_{x,x'}\frac{1}{N^2}\left(f(x)^\top f(x')\right)^2 \tag{14}$$

$$= \left\|\frac{N}{|E|}\cdot A - \left(\frac{1}{\sqrt{N}}\cdot F\right)\left(\frac{1}{\sqrt{N}}\cdot F\right)^\top\right\|_F^2 + \text{const}, \tag{15}$$

where $F \in \mathbb{R}^{N\times k}$ is the matrix which the x-th row contains $f(x)^\top$. According to the Eckart–Young–Mirsky theorem, the minimizer of this loss function is $F = \frac{N}{\sqrt{|E|}}S_k$ where $S_k \in \mathbb{R}^{N\times k}$ is a matrix such that $A_k = S_k^\top S_k$.

Let $S_{k,\mathcal{S}} \in \mathbb{R}^{|\mathcal{S}|\times k}$ be the matrix gotten by restricting S_k to the rows corresponding to the source data, and $S_{k,\mathcal{T}}$ be the matrix gotten by restricting S_k to the rows corresponding to the target data. The head learned on the source domain can be expressed as

$$\hat{h} = \operatorname*{argmin}_{h\in\mathbb{R}^{(k\times r)}}\sum_{x\in\mathcal{S}}\left(\left\|h^\top f(x) - \vec{y}_x\right\|_2^2 + \eta\|h\|_F^2\right) \tag{16}$$

$$= \frac{\sqrt{|E|}}{N}\cdot S_{k,\mathcal{S}}^\top\left(S_{k,\mathcal{S}}S_{k,\mathcal{S}}^\top + \frac{|E|}{N^2}\cdot\eta\cdot|S|\cdot I\right)^\dagger Y_\mathcal{S}. \tag{17}$$

Therefore, the prediction on the target domain $\text{pred}_\mathcal{T}$ is

$$\text{pred}_\mathcal{T} = F_\mathcal{T}\hat{h} = S_{k,\mathcal{T}}S_{k,\mathcal{S}}^\top\left(S_{k,\mathcal{S}}S_{k,\mathcal{S}}^\top + \frac{|E|}{N^2}\cdot\eta\cdot|S|\cdot I\right)^\dagger Y_\mathcal{S} = A_{k,(\mathcal{T},\mathcal{S})}\left(A_{k,(\mathcal{S},\mathcal{S})} + \frac{|E|}{N^2}\cdot\eta\cdot|S|\cdot I\right)^\dagger Y_\mathcal{S}. \tag{18}$$

□

The following lemma shows that the prediction in the expectation graph is accuracte.

Lemma 2. *Let $\tilde{A} \triangleq \mathbb{E}[A]$ be the expectation of the adjacency matrix. Then, for any $\xi > 0$, we have*

$$\tilde{A}_{k,(\mathcal{T},\mathcal{S})}\left(\tilde{A}_{k,(\mathcal{S},\mathcal{S})} + \xi I\right)^\dagger Y_\mathcal{S} = \frac{\lambda_c}{\lambda_c + m\xi}\cdot Y_\mathcal{T}, \tag{19}$$

where $k = r + m - 1$ and $\lambda_c \triangleq n\rho - n\beta + n(m-1)\alpha - n(m-1)\gamma$. Furthermore, if we use $\tilde{\lambda}_i$ to denote the i-th largest eigenvalue of $\tilde{A}$, we have $\tilde{\lambda}_k - \tilde{\lambda}_{k+1} = n\min\{r(\beta-\gamma), m(\alpha-\gamma)\}$ and $\tilde{\lambda}_1 \leq nrm\rho$.

Proof of Lemma 2. By the definition of the graph, every entry $\tilde{A}_{xx'}$ is in the set of $\{\rho, \beta, \alpha, \gamma\}$, depending on whether x and x' belong to the same domain/class. We can index every node x as (d_x, y_x, id_x), where $\text{id}_x \in [n]$ is the index of x within domain d_x and class y_x. For any integer $i \geq 1$, we use $\mathbf{1}_i$ to denote the i-dimensional all-one vector, and $\bar{\mathbf{1}}_i = \mathbf{1}_i/\|\mathbf{1}_i\|$ be its normalized unit vector. We use $\mathbb{S}^i$ to denote the i-dimensional unit-norm sphere.

It can be verified that $\tilde{A}$ can be decomposed into the sum of several matrix Kronecker products:

$$\begin{aligned}\tilde{A} = \; & (\beta-\gamma)\cdot I_m\otimes(\mathbf{1}_r\mathbf{1}_r^\top)\otimes(\mathbf{1}_n\mathbf{1}_n^\top) && (20)\\ & +(\alpha-\gamma)\cdot(\mathbf{1}_m\mathbf{1}_m^\top)\otimes I_r\otimes(\mathbf{1}_n\mathbf{1}_n^\top) && (21)\\ & +(\rho-\beta-\alpha+\gamma)\cdot I_m\otimes I_r\otimes(\mathbf{1}_n\mathbf{1}_n^\top) && (22)\\ & +\gamma\cdot(\mathbf{1}_m\mathbf{1}_m^\top)\otimes(\mathbf{1}_r\mathbf{1}_r^\top)\otimes(\mathbf{1}_n\mathbf{1}_n^\top). && (23)\end{aligned}$$

As a result, $\tilde{A}$ has the following four sets of eigenvectors with non-zero eigenvalues:

- $\bar{\mathbf{1}}_m\otimes\bar{\mathbf{1}}_r\otimes\bar{\mathbf{1}}_n$. The corresponding eigenvalue is $\lambda_a \triangleq n\rho+n(r-1)\beta+n(m-1)\alpha+n(m-1)(r-1)\gamma$.
- $u\otimes\bar{\mathbf{1}}_r\otimes\bar{\mathbf{1}}_n$, where $u\in\mathbb{S}^m$ and $u^\top\mathbf{1}_m=0$. The corresponding eigenvalue is $\lambda_b\triangleq n\rho+n(r-1)\beta-n\alpha-n(r-1)\gamma$.
- $\bar{\mathbf{1}}_m\otimes v\otimes\bar{\mathbf{1}}_n$, where $v\in\mathbb{S}^r$ and $v^\top\mathbf{1}_r=0$. The corresponding eigenvalue is $\lambda_c\triangleq n\rho-n\beta+n(m-1)\alpha-n(m-1)\gamma$.
- $u\otimes v\otimes\bar{\mathbf{1}}_n$, where $u\in\mathbb{S}^m$, $v\in\mathbb{S}^r$ and $u^\top\mathbf{1}_m=0$, $v^\top\mathbf{1}_r=0$. The corresponding eigenvalue is $\lambda_d\triangleq n\rho-n\beta-n\alpha+n\gamma$.

Since $\rho>\max\{\beta,\alpha\}$ and $\min\{\beta,\alpha\}>\gamma$, we know that all of these eigenvalues are positive. When $k=r+m-1$, $\tilde{A}_k$ will contain exactly the first three sets of eigenvectors since they correspond to the top-k eigenvalues. This suggests that we can write $\tilde{A}_k$ as follows

$$\tilde{A}_k=\lambda_a\cdot\bar{\mathbf{1}}_m\bar{\mathbf{1}}_m^\top\otimes\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top+\lambda_b\cdot(I_m-\bar{\mathbf{1}}_m\bar{\mathbf{1}}_m^\top)\otimes\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top+\lambda_c\cdot\bar{\mathbf{1}}_m\bar{\mathbf{1}}_m^\top\otimes(I_r-\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top)\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top. \tag{24}$$

Restricting to the source domain, we have

$$\tilde{A}_{k,(\mathcal{S},\mathcal{S})}=\frac{\lambda_a+(m-1)\lambda_b}{m}\cdot\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top+\frac{\lambda_c}{m}\cdot(I_r-\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top)\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top. \tag{25}$$

By the definition of pseudoinverse, we have

$$\left(\tilde{A}_{k,(\mathcal{S},\mathcal{S})}+\xi I\right)^\dagger=\left(\frac{\lambda_a+(m-1)\lambda_b}{m}+\xi\right)^{-1}\cdot\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top+\left(\frac{\lambda_c}{m}+\xi\right)^{-1}\cdot(I_r-\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top)\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top. \tag{26}$$

Notice that $Y_\mathcal{S}$ satisfies $\left(\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top\right)Y_\mathcal{S}=0$ and $\left((I_r-\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top)\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top\right)Y_\mathcal{S}=Y_\mathcal{S}$, we have

$$\left(\tilde{A}_{k,(\mathcal{S},\mathcal{S})}+\xi I\right)^\dagger Y_\mathcal{S}=\left(\frac{\lambda_c}{m}+\xi\right)^{-1}Y_\mathcal{S}. \tag{27}$$

We can also write $\tilde{A}_{k,(\mathcal{X},\mathcal{S})}$ in the form of kronecker products as follows:

$$\tilde{A}_{k,(\mathcal{X},\mathcal{S})}=\frac{\lambda_a}{m}\cdot\mathbf{1}_m\otimes\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top+\lambda_b\cdot(e_1-\frac{1}{m}\mathbf{1}_m)\otimes\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top+\frac{\lambda_c}{m}\cdot\mathbf{1}_m\otimes(I_r-\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top)\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top. \tag{28}$$

Again, using the fact that $\left(\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top\right)Y_\mathcal{S}=0$ and $\left((I_r-\bar{\mathbf{1}}_r\bar{\mathbf{1}}_r^\top)\otimes\bar{\mathbf{1}}_n\bar{\mathbf{1}}_n^\top\right)Y_\mathcal{S}=Y_\mathcal{S}$, we have

$$\tilde{A}_{k,(\mathcal{X},\mathcal{S})}\left(\tilde{A}_{k,(\mathcal{S},\mathcal{S})}+\xi I\right)^\dagger Y_\mathcal{S}=\frac{\lambda_c}{\lambda_c+m\xi}\mathbf{1}_m\otimes Y_\mathcal{S}. \tag{29}$$

Finally, noticing that $\mathbf{1}_m\otimes Y_\mathcal{S}=Y$ finishes the proof.

□

The followng lemma shows that when a matrix A is close to $\tilde{A}$, their rank-k approximations A_k and $\tilde{A}_k$ are also close.

Lemma 3. *Let $\tilde{A} \triangleq \mathbb{E}[A]$ be the expectation of the adjacency matrix. Let A_k and $\tilde{A}_k$ be the rank-k approximations of A and $\tilde{A}$, respectively. Let $\tilde{\lambda}_i$ be the i-th largest eigenvalue of $\tilde{A}$, $\|\cdot\|$ be the operator norm of a matrix or ℓ_2-norm of a vector. Then, when $\left\|A-\tilde{A}\right\| < \tilde{\lambda}_k - \tilde{\lambda}_{k+1}$, we have*

$$\left\|A_k-\tilde{A}_k\right\| \leq \left(1+\frac{2\left\|A-\tilde{A}\right\|+2\left\|\tilde{A}\right\|}{(\tilde{\lambda}_k-\tilde{\lambda}_{k+1})-\left\|A-\tilde{A}\right\|}\right)\cdot\left\|A-\tilde{A}\right\|. \tag{30}$$

Proof of Lemma 3. Let the SVD decomposition of A and $\tilde{A}$ be as follows:

$$A = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} U_1^\top \\ U_2^\top \end{bmatrix}, \tag{31}$$

$$\tilde{A} = \begin{bmatrix} \tilde{U}_1 & \tilde{U}_2 \end{bmatrix} \begin{bmatrix} \tilde{\Sigma}_1 & 0 \\ 0 & \tilde{\Sigma}_2 \end{bmatrix} \begin{bmatrix} \tilde{U}_1^\top \\ \tilde{U}_2^\top \end{bmatrix}, \tag{32}$$

where Σ_1 and $\tilde{\Sigma}_1$ contain the top k eigenvalues of A and $\tilde{A}$, respectively. By the definition of rank-k approximation, we have $A_k = U_1\Sigma_1U_1^\top$ and $\tilde{A}_k = \tilde{U}_1\tilde{\Sigma}_1\tilde{U}_1^\top$. Therefore,

$$\left\|A_k-\tilde{A}_k\right\| = \left\|U_1\Sigma_1U_1^\top - \tilde{U}_1\tilde{\Sigma}_1\tilde{U}_1^\top\right\| = \max_{v\in\mathbb{R}^N:\|v\|=1}\left\|\left(U_1\Sigma_1U_1^\top - \tilde{U}_1\tilde{\Sigma}_1\tilde{U}_1^\top\right)v\right\| \tag{33}$$

$$\leq \underbrace{\max_{v\in\mathbb{R}^N:\|v\|=1, v^\top\tilde{U}_2=0}\left\|\left(U_1\Sigma_1U_1^\top - \tilde{U}_1\tilde{\Sigma}_1\tilde{U}_1^\top\right)v\right\|}_{C_1} + \underbrace{\max_{v\in\mathbb{R}^N:\|v\|=1, v^\top\tilde{U}_1=0}\left\|\left(U_1\Sigma_1U_1^\top - \tilde{U}_1\tilde{\Sigma}_1\tilde{U}_1^\top\right)v\right\|}_{C_2}. \tag{34}$$

Let $\delta \triangleq \min\{\lambda_k - \tilde{\lambda}_{k+1}, \tilde{\lambda}_k - \lambda_{k+1}\}$. According to Weyl's inequality, we have

$$\left\|\begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} - \begin{bmatrix} \tilde{\Sigma}_1 & 0 \\ 0 & \tilde{\Sigma}_2 \end{bmatrix}\right\| \leq \left\|A-\tilde{A}\right\|. \tag{35}$$

So we have $\delta \geq \left(\tilde{\lambda}_k - \tilde{\lambda}_{k+1}\right) - \left\|A-\tilde{A}\right\| > 0$. Now we bound C_1 and C_2 separately. To bound C_1, we have

$$C_1 \leq \left\|A-\tilde{A}\right\| + \max_{v\in\mathbb{R}^N:\|v\|=1, v^\top\tilde{U}_2=0}\left\|\left(U_2\Sigma_2U_2^\top - \tilde{U}_2\tilde{\Sigma}_2\tilde{U}_2^\top\right)v\right\| \tag{36}$$

$$\leq \left\|A-\tilde{A}\right\| + \|U_2\|\cdot\|\Sigma_2\|\cdot\left\|U_2^\top\tilde{U}_1\right\| \tag{37}$$

$$\leq \left(1+\frac{\|\Sigma_2\|}{\delta}\right)\left\|A-\tilde{A}\right\| \tag{38}$$

where the first line follows the triangle inequality, the second line uses $v^\top\tilde{U}_2 = 0$, and the third line uses Davis-Kahan theorem (Davis & Kahan, 1970). Similarly, to bound C_2, we have

$$C_2 = \max_{v\in\mathbb{R}^N:\|v\|=1, v^\top\tilde{U}_1=0}\left\|U_1\Sigma_1U_1^\top v\right\|_2 \tag{39}$$

$$\leq \|U_1\|\cdot\|\Sigma_2\|\cdot\left\|U_1^\top\tilde{U}_2\right\| \leq \frac{\|\Sigma_1\|\cdot\left\|A-\tilde{A}\right\|}{\delta}. \tag{40}$$

Adding C_1 and C_2 together we have

$$\left\|A_k-\tilde{A}_k\right\| \leq \left(1+\frac{\|\Sigma_1\|+\|\Sigma_2\|}{\delta}\right)\cdot\left\|A-\tilde{A}\right\| \tag{41}$$

$$\leq \left(1+\frac{2\left\|A-\tilde{A}\right\|+2\left\|\tilde{A}\right\|}{(\tilde{\lambda}_k-\tilde{\lambda}_{k+1})-\left\|A-\tilde{A}\right\|}\right)\cdot\left\|A-\tilde{A}\right\|, \tag{42}$$

where the second line again uses Weyl's inequality. □

To bound the differene between A and $\tilde{A}$, we use the following lemma which is adapted from Theorem 5.2 of (Lei & Rinaldo, 2015).

Lemma 4. *Let A be the adjacency matrix of a random graph on N nodes in which edges occur independently. Let $E[A]=\tilde{A}=(\tilde{A}_{ij})_{i,j=1,\ldots,N}$ be the expectation adjacency matrix and assume that $N\max_{ij}\tilde{A}_{ij}\geq \log N$. Then, for any $\zeta>0$ there exists a constant $C=C(\zeta)$ such that*

$$\|A-P\|\leq C\sqrt{N} \tag{43}$$

with probability at least $1-N^{-2\zeta}$.

Now we prove Theorem 4 using the above lemmas.

Proof of Theorem 4. By Lemma 1, we have that the prediction on the target domain is

$$\text{pred}_{\mathcal{T}}=A_{k,(\mathcal{T},\mathcal{S})}\left(A_{k,(\mathcal{S},\mathcal{S})}+\frac{|E|}{N^2}\cdot\eta\cdot|S|\cdot I\right)^{\dagger}Y_{\mathcal{S}}. \tag{44}$$

Let $|\tilde{E}|\triangleq \mathbf{1}_N^{\top}\tilde{A}\mathbf{1}_N$ be the expectation of number of edges in the graph. We define the ideal prediction as

$$\widetilde{\text{pred}}_{\mathcal{T}}=\tilde{A}_{k,(\mathcal{T},\mathcal{S})}\left(\tilde{A}_{k,(\mathcal{S},\mathcal{S})}+\frac{|\tilde{E}|}{N^2}\cdot\eta\cdot|S|\cdot I\right)^{\dagger}Y_{\mathcal{S}}. \tag{45}$$

We will bound the difference between $\text{pred}_{\mathcal{T}}$ and $\widetilde{\text{pred}}_{\mathcal{T}}$. For every class $c\in[r]$, define the following error vector

$$e^c\triangleq A_{k,(\mathcal{T},\mathcal{S})}\left(A_{k,(\mathcal{S},\mathcal{S})}+\frac{|E|}{N^2}\cdot\eta\cdot|S|\cdot I\right)^{\dagger}Y_{\mathcal{S}}^c-\tilde{A}_{k,(\mathcal{T},\mathcal{S})}\left(\tilde{A}_{k,(\mathcal{S},\mathcal{S})}+\frac{|E|}{N^2}\cdot\eta\cdot|S|\cdot I\right)^{\dagger}Y_{\mathcal{S}}^c, \tag{46}$$

where $Y_{\mathcal{S}}^c$ is the c-th column of $Y_{\mathcal{S}}$.

Using the perturbation bound for the pseudoinverse matrix (Stewart, 1977), we have

$$\|e^c\|\leq\left\|A_{k,(\mathcal{T},\mathcal{S})}-\tilde{A}_{k,(\mathcal{T},\mathcal{S})}\right\|\cdot\left\|\left(\tilde{A}_{k,(\mathcal{S},\mathcal{S})}+\frac{|E|}{N^2}\cdot\eta\cdot|S|\cdot I\right)^{\dagger}\right\|\cdot\|Y_{\mathcal{S}}^c\| \tag{47}$$

$$+\|A_{k,(\mathcal{T},\mathcal{S})}\|\cdot\left\|\left(A_{k,(\mathcal{S},\mathcal{S})}+\frac{|E|}{N^2}\cdot\eta\cdot|S|\cdot I\right)^{\dagger}-\left(\tilde{A}_{k,(\mathcal{S},\mathcal{S})}+\frac{|E|}{N^2}\cdot\eta\cdot|S|\cdot I\right)^{\dagger}\right\|\cdot\|Y_{\mathcal{S}}^c\| \tag{48}$$

$$\leq\left\|A_k-\tilde{A}_k\right\|\cdot\frac{N^2}{\eta|E|\cdot|S|}\cdot\|Y_{\mathcal{S}}^c\|+\|A_k\|\cdot\frac{1+\sqrt{5}}{2}\cdot\left(\frac{N^2}{\eta|E|\cdot|S|}\right)^2\cdot\left\|A_k-\tilde{A}_k\right\|\cdot\|Y_{\mathcal{S}}^c\|. \tag{49}$$

By lemma 4, there exists constant $C=C(\zeta)$ such that $\left\|A-\tilde{A}\right\|\leq C\sqrt{N}$ with probability at least $1-N^{-2\zeta}$ for any $N>\Omega(1/\rho)$. From now on, we assume this high probability event happens. Let $\Delta\triangleq\min\{r(\beta-\gamma),m(\alpha-\gamma)\}$. According to Lemma 2, we know that $\tilde{\lambda}_k-\tilde{\lambda}_{k+1}=n\Delta$. If our choice of N further satisfies $N\geq\left(\frac{2rmC}{\Delta}\right)^2$, we have $\left\|A-\tilde{A}\right\|\leq\frac{1}{2}(\tilde{\lambda}_k-\tilde{\lambda}_{k+1})$, so from Lemma 3 we have

$$\left\|A_k-\tilde{A}_k\right\|\leq O\left(\frac{\tilde{\lambda}_1}{\tilde{\lambda}_k-\tilde{\lambda}_{k+1}}\cdot\left\|A-\tilde{A}\right\|\right)\leq O\left(\frac{\tilde{\lambda}_1}{\tilde{\lambda}_k-\tilde{\lambda}_{k+1}}\cdot\sqrt{N}\right). \tag{50}$$

According to Lemma 2, we know that $\tilde{\lambda}_1\leq nrm\rho$, so we have

$$\left\|A_k-\tilde{A}_k\right\|\leq O\left(\frac{rm\rho\sqrt{N}}{\Delta}\right). \tag{51}$$

By Hoeffding's inequality, with probability at least $1-2e^{-2N^2}$ we have $||E|-|\tilde{E}|| \leq N$. From now on, we assume this high-probability event happens. The total failure probability so far is $N^{-2\zeta}+2e^{-2N^2} \leq N^{-\zeta}$. By the definition of graph, we have $|\tilde{E}| \geq \frac{\rho N^2}{rm}$. If our choice of N further satisfies $N \geq \frac{2rm}{\rho}$, we have $|E| \geq \frac{\rho N^2}{2rm}$, hence

$$\frac{|E| \cdot |S|}{N^2} \geq \frac{\rho N}{2rm^2}. \tag{52}$$

Substituting (51) and (52) into (49) gives:

$$\|e^c\| \leq O\left(\frac{1}{\Delta\eta^2\sqrt{N}}\right) \cdot \mathrm{poly}(r,m) \cdot \|Y_{\mathcal{S}}^c\|. \tag{53}$$

Summing over all classes c and noticing that $\|Y_{\mathcal{S}}^c\| \leq \sqrt{N}$ leads to

$$\|e\|_F \leq O\left(\frac{1}{\Delta\eta^2}\right) \cdot \mathrm{poly}(r,m) \tag{54}$$

Let $\xi = \frac{|E|}{N^2} \cdot \eta \cdot |S|$ and $\tilde{\xi} = \frac{|\tilde{E}|}{N^2} \cdot \eta \cdot |S|$. We have

$$\left|\frac{\lambda_c}{\lambda_c+m\xi} - \frac{\lambda_c}{\lambda_c+m\tilde{\xi}}\right| \leq \frac{m}{\lambda_c} \cdot |\xi-\tilde{\xi}| \leq \frac{\eta}{N} \cdot \mathrm{poly}(r,m). \tag{55}$$

From Lemma 2 we have

$$\left\|\tilde{A}_{k,(\mathcal{T},\mathcal{S})}\left(\tilde{A}_{k,(\mathcal{S},\mathcal{S})}+\frac{|E|}{N^2} \cdot \eta \cdot |S| \cdot I\right)^{\dagger} Y_{\mathcal{S}} - \tilde{A}_{k,(\mathcal{T},\mathcal{S})}\left(\tilde{A}_{k,(\mathcal{S},\mathcal{S})}+\frac{|\tilde{E}|}{N^2} \cdot \eta \cdot |S| \cdot I\right)^{\dagger} Y_{\mathcal{S}}\right\|_F \tag{56}$$

$$= \left|\frac{\lambda_c}{\lambda_c+m\xi} - \frac{\lambda_c}{\lambda_c+m\tilde{\xi}}\right| \cdot \|Y_{\mathcal{T}}\|_F \leq \frac{1}{\sqrt{N}} \cdot \mathrm{poly}(r,m). \tag{57}$$

Combining (54) and (57) we have

$$\left\|\mathrm{pred}_{\mathcal{T}} - \widetilde{\mathrm{pred}}_{\mathcal{T}}\right\|_F \leq O\left(\frac{1}{\eta^2 \cdot \min\{\alpha-\gamma,\beta-\gamma\}}\right) \cdot \mathrm{poly}(r,m). \tag{58}$$

Notice that the x-th row of $\widetilde{\mathrm{pred}}_{\mathcal{T}}$ is $\frac{\lambda_c}{\lambda_c+m\tilde{\xi}} \cdot y_{y_x}$. Since $\lambda_c = n\left(\rho-\beta+(m-1)\alpha-(m-1)\gamma\right) \geq \frac{1}{2}nm(\alpha-\gamma)$, and $\tilde{\xi} = \frac{|\tilde{E}|}{N^2} \cdot \eta \cdot |S| = \frac{n^2rm\rho+n^2m(r^2-r)\beta+n^2r(m^2-m)\alpha+n^2(r^2-r)(m^2-m)\gamma}{n^2m^2r^2} \cdot \eta \cdot rn \leq \eta rn\rho$, we have

$$\frac{\lambda_c}{\lambda_c+m\tilde{\xi}} \geq \frac{1}{1+\frac{2r\rho\eta}{m(\alpha-\gamma)}} \geq 1-\epsilon, \tag{59}$$

where the second inequality follows our assumption on η.

Since $\mathrm{pred}_{\mathcal{T}}$ is incorrect on the x-th row only if its difference from the x-th row of $\widetilde{\mathrm{pred}}_{\mathcal{T}}$ has larger norm than $\Omega(1-\epsilon)$, we know the final total error on the target domains is bounded by $O\left(\frac{1}{\eta^4 N \cdot \min\{\alpha-\gamma,\beta-\gamma\}^2}\right) \cdot \mathrm{poly}(r,m)$.

Collecting all the requirements of N, this bound holds so long as $N \geq \Omega\left(\left(\frac{rm}{\min\{\alpha-\gamma,\beta-\gamma\}}\right)^2\right)$, which is equivalent to $n \geq \Omega\left(\frac{rm}{\min\{\alpha-\gamma,\beta-\gamma\}^2}\right)$.

□

Proof of Proposition 2. Notice that the roles of domain and class are the same in stochastic block model. Let $\zeta > 0$ and $\epsilon \in \left(0, \frac{1}{2}\right)$ be arbitrary constants. If we train a domain classifier head on class 1 with regularization strength $\eta \in \left(0, \frac{(\beta-\gamma)\epsilon}{2r\rho}\right]$, then following the same proof of Theorem 4, we know that when $n \geq \Omega\left(\frac{rm}{\min\{\alpha-\gamma,\beta-\gamma\}^2}\right)$, with probability at least $1-n^{-\zeta}$, we have that the global error of this domain predictor is bounded by $O\left(\frac{1}{\eta^4 \cdot \min\{\alpha-\gamma,\beta-\gamma\}^2 \cdot n}\right) \cdot \mathrm{poly}(r,m)$. Setting $\epsilon = \frac{1}{4}$, $\zeta = 1$ and $\eta = \frac{\beta-\gamma}{8r\rho}$ finishes the proof. □

B.2. Proof of Proposition 3

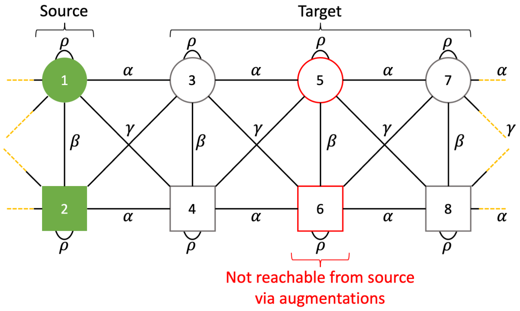

Figure 4: Example distribution of data and augmentations in which ERM and DANN generalize more poorly than contrastive learning. ERM and DANN minimizers are only guaranteed to correctly label inputs that are either in the source domain or reachable from source inputs via data augmentation, while connectivity throughout the target domain allows contrastive pre-training to generalize even to target inputs that are multiple "hops" away from the source. The orange dotted lines on the far left and far right connect to each other but are illustrated as straight dotted lines for cleanliness.

Data distribution (Figure 4). In the setting of binary classification with 2 domains, let $\mathcal{S} = \{x \in \mathcal{X} : d_x = 1\}$ and $\mathcal{T} = \{x \in \mathcal{T} : d_x = 2\}$ and assume that P_S and P_T are uniform over $\mathcal{S}$ and $\mathcal{T}$, respectively. We refer to each input by its ID, so that $\mathcal{X} = \{1,...,8\}$. The source domain $\mathcal{S} = \{1,2\}$ contains 2 points while the target domain $\mathcal{T} = \{3,4,5,6,7,8\}$ contains 6 points. The label space is $\mathcal{Y} = \{2,1\}$. The label for $x \in \{1,3,5,7\}$ is $y_x = 1$ and the label for $x \in \{2,4,6,8\}$ is $y_x = 2$. The marginal distribution over unlabeled examples P_U (used by contrastive pre-training in Equation 4) is the uniform distribution over $\mathcal{X}$. Only the inputs in the source domain $\mathcal{S}$ have labels for supervised learning.

Augmentation distribution. The augmentation distribution $\mathcal{A}(\cdot \mid x)$ is

$$\mathcal{A}(x' \mid x) = \begin{cases} \rho' & x = x' \\ \alpha' & y_{x'} = y_x, x \neq x' \\ \beta' & \{x',x\} \in \{\{1,2\},\{3,4\},\{5,6\},\{7,8\}\} \\ \gamma' & \{x',x\} \in \{\{1,4\},\{2,3\},\{3,6\},\{4,5\},\{5,8\},\{6,7\},\{1,8\},\{2,7\}\} \end{cases} . \tag{60}$$

By the edge structure of the graph, we have that $\rho' + 2\alpha' + \beta' + 2\gamma' = 1$. We assume that ρ', α', β', and γ' are all distinct and obey the following inequalities: $\rho' > \max\{\alpha', \beta'\}$ and $\min\{\alpha', \beta'\} > \gamma'$.

Distribution over positive pairs. The augmentation distribution induces a distribution S_+ over positive pairs, which is defined similarly:

$$S_+(x',x) = \begin{cases} \rho/C & x = x' \\ \alpha/C & y_{x'} = y_x, x \neq x' \\ \beta/C & \{x',x\} \in \{\{1,2\},\{3,4\},\{5,6\},\{7,8\}\} \\ \gamma/C & \{x',x\} \in \{\{1,4\},\{2,3\},\{3,6\},\{4,5\},\{5,8\},\{6,7\},\{1,8\},\{2,7\}\} \end{cases} \tag{61}$$

where the normalization is $C = 8\rho + 16\alpha + 8\beta + 16\gamma$. The normalization ensures that the sum of $S_+(x',x)$ over all pairs x',x is 1. At the end of this section (paragraph B.2), we show how to derive ρ,α,β,γ as functions of $\rho',\alpha',\beta',\gamma'$, respectively. We also show that the assumptions that $\rho' > \max\{\alpha',\beta'\}$ and $\min\{\alpha',\beta'\} > \gamma'$ imply that $\rho > \max\{\alpha,\beta\}$ and $\min\{\alpha,\beta\} > \gamma$.

We recall Proposition 3:

Proposition 3. *There exists a set $\mathcal{X}$, a distribution P_U and data augmentation $\mathcal{A}$, such that for some feature dimension $k \in \mathbb{Z}^+$, a linear classifier trained on contrastive pre-trained features achieves 0 target error: $\mathcal{L}_{0-1}(\widehat{f}) = 0$. However, for all $k \in \mathbb{Z}^+$, there exist minimizers $\widehat{f}_{erm}$ and $\widehat{f}_{dann}$ of the ERM and DANN objectives, respectively, that have non-zero error: $\mathcal{L}_{0-1}(\widehat{f}_{erm}) = \mathcal{L}_{0-1}(\widehat{f}_{dann}) = 1/3$.*

ERM. We claim that the a classifier $\widehat{f}_{\text{erm}}$ that makes the following predictions is a minimizer of the ERM objective with data augmentation (Equation 1):

$$\underset{i}{\operatorname{argmax}} \widehat{f}_{\text{erm}}(x)_i = \begin{cases} 1 & \text{if} \quad x \in \{1,3,6,7\} \\ 2 & \text{if} \quad x \in \{2,4,5,8\}. \end{cases} \tag{62}$$

Since this classifier outputs the incorrect prediction 2/6 of the target points ($\{5,6\}$), it has a target error of 1/3. It remains to show that $\widehat{f}_{\text{erm}}$ is a minimizer of the ERM objective.

To see why, we first define the reachable set $\mathcal{R} = \{1,2,3,4,7,8\}$ as the inputs that are sampled with probability greater than 0 as augmentations of the labeled source data, and rewrite Equation 1 as

$$\mathcal{L}_{\text{ERM}}(f) = \mathbb{E}_{x \sim P_S, x' \sim \mathcal{A}(\cdot|x)}[\mathbf{1}[\underset{i}{\operatorname{argmax}} f(x')_i \neq y_x]] \tag{63}$$

$$= \sum_{x \in \{1,2\}} P_S(x) \sum_{x' \in \mathcal{X}} \mathcal{A}(x' \mid x)[\mathbf{1}[\underset{i}{\operatorname{argmax}} f(x')_i \neq y_x]] \tag{64}$$

$$= \sum_{x \in \{1,2\}} P_S(x) \sum_{x' \in \mathcal{R}} \mathcal{A}(x' \mid x)[\mathbf{1}[\underset{i}{\operatorname{argmax}} f(x')_i \neq y_x]] \tag{65}$$

where in the last step, we replace the sum over the entire input space $\mathcal{X}$ with the reachable set $\mathcal{R}$ since $\mathcal{A}(x' \mid x) = 0$ when x' is not in the reachable set. Since $x \in \{5,6\}$ are not included in this sum, the prediction of f on $\{5,6\}$ can be arbitrary (and wrong). This shows that there exists a minimizer of the ERM objective with error at least 1/3, since the minimizer can classify 2/6 target inputs incorrectly.

However, all minimizers of ERM output the correct prediction for the other inputs. From above, we have

$$\mathcal{L}_{\text{ERM}}(f) = P_S(x=1)\left(\sum_{x' \in \mathcal{R}} \mathcal{A}(x' \mid x=1)\mathbf{1}[\underset{i}{\operatorname{argmax}} f(x')_i \neq 1]\right) + P_S(x=2)\left(\sum_{x' \in \mathcal{R}} \mathcal{A}(x' \mid x=2)\mathbf{1}[\underset{i}{\operatorname{argmax}} f(x')_i \neq 2]\right) \tag{66}$$

$$= \frac{1}{2}\sum_{x' \in \mathcal{R}} \mathcal{A}(x' \mid x=1) \cdot \mathbf{1}[\underset{i}{\operatorname{argmax}} f(x')_i \neq 1] + \mathcal{A}(x' \mid x=2) \cdot \mathbf{1}[\underset{i}{\operatorname{argmax}} f(x')_i \neq 2] \tag{67}$$

since $P_S(x=1) = P_S(x=2) = 1/2$. For each $x' \in \mathcal{R}$, the minimizer will output 1 if $\mathcal{A}(x' \mid x=1) > \mathcal{A}(x' \mid x=2)$. For $x' = 1$, we have that $\mathcal{A}(x' \mid x=1) = \rho'$, which by assumption is larger than $\mathcal{A}(x' \mid x=2) = \beta'$. For $x' \in \{3,7\}$, we have that $\mathcal{A}(x' \mid x=1) = \alpha'$, which by assumption is larger than $\mathcal{A}(x' \mid x=2) = \gamma'$. Therefore any minimizer of the ERM objective will output 1 for $x \in \{1,3,7\}$. By symmetry, any minimizer of the ERM objective will output 2 for $x \in \{2,4,8\}$. Thus any minimizer of the ERM objective will correctly classify these 6 inputs (including 4/6 of the target inputs). This shows that $\widehat{f}_{\text{erm}}$ is a minimizer of the ERM objective and has target error 1/3.

DANN. Let $z_1, z_2 \in \mathbb{R}^k$ be distinct points in the representation space such that $z_1 \neq z_2$. We claim that the following encoder $\widehat{\phi}_{\text{dann}}$ and classification head $\widehat{h}_{\text{dann}}$ minimize the DANN objective (Equation 2):

$$\widehat{\phi}_{\text{dann}}(x) = \begin{cases} z_1 & \text{if} \quad x \in \{1,3,6,7\} \\ z_2 & \text{if} \quad x \in \{2,4,5,8\} \end{cases} \quad \text{and} \quad \underset{i}{\operatorname{argmax}} \widehat{h}_{\text{dann}}(z)_i = \begin{cases} 1 & \text{if} \quad z = z_1 \\ 2 & \text{if} \quad z = z_2 \end{cases}. \tag{68}$$

The first term in the DANN objective is equivalent to the ERM objective, and the classifier $\widehat{f}_{\text{dann}} : \widehat{h}_{\text{dann}} \circ \widehat{\phi}_{\text{dann}}$ outputs the same predictions as the ERM minimizer $\widehat{f}_{\text{erm}}$. Therefore, $\widehat{f}_{\text{dann}}$ minimizes the first term of Equation 2 by the same argument from the ERM section.

We show that $\widehat{\phi}_{\text{dann}}$ minimizes the second term of Equation 2. First, we show that the second term is lower bounded by $-\lambda/4$. To do this, we first consider the distribution of source inputs after augmentation. The probability that an augmentation x' of $x \sim P_U$ is in $\mathcal{S} = \{1,2\}$ is

$$\begin{aligned}\mathbb{P}[x' \in \mathcal{S}] &= \sum_{x \in \mathcal{X}} P_U(x) \sum_{x' \in \mathcal{S}} \mathcal{A}(x' \mid x) && (69)\\ &= 2 \cdot \frac{1}{8}(\rho' + \beta') + 4 \cdot \frac{1}{8}(\alpha' + \gamma') && (70)\\ &= \frac{1}{4}(\rho' + \beta' + 2\alpha' + 2\gamma') && (71)\\ &= 1/4. && (72)\end{aligned}$$

Let $h_d : \mathbb{R}^k \to \mathbb{R}^2$ be a domain classification head (from the model family of all functions from $\mathbb{R}^k$ to $\mathbb{R}^2$). For any encoder ϕ (from the model family of all functions from $\mathcal{X}$ to $\mathbb{R}^k$), consider a domain classification head h_d which always predicts target, so that $h_d(\phi(x')) = 2$ for all x'. This domain classification head attains error $1/4$ (for the misclassified source inputs). Therefore the second term of the DANN loss is lower bounded (i.e., the domain classification error is upper bounded):

$$\begin{aligned}&-\min_{h_d} \lambda \cdot \mathbb{E}_{x \sim P_U, x' \sim \mathcal{A}(\cdot \mid x)}[\mathbf{1}[\operatorname*{argmax}_i h_d(\phi(x'))_i \neq d_x]] && (73)\\ &\geq -\lambda/4. && (74)\end{aligned}$$

where λ is the weight of the second term in the DANN loss.

Next, we show that $\widehat{\phi}_{\text{dann}}$ attains the optimal value $-\lambda/4$. Since $\widehat{\phi}_{\text{dann}}(x')$ can only be one of two values, it is straightforward to see that, the minimizing domain classification head h_d must predict $\operatorname{argmax}_i h_d(z_1)_i = 1$ and $\operatorname{argmax}_i h_d(z_2)_i = 2$. This shows that $\widehat{\phi}_{\text{dann}}$ achieves the optimal value of $-\lambda/4$ for the second term.

Contrastive pre-training. We show that contrastive pre-training achieves 0 target error on this example as long as $\alpha > \gamma + \beta$. Examining Figure 4, α intuitively corresponds to the probability that an image x augments to a different image x' of the same class, while γ and β correspond to the probability that x augments to x' of a different class. Roughly speaking, the condition $\alpha > \gamma + \beta$ means that augmentations should preserve the class more often than not.

We prove that contrastive pre-training achieves 0 target error by first deriving the representations learned by contrastive learning on unlabeled data P_U (Equation 6), and then deriving the linear probe learned by optimizing the squared loss on source examples $\mathcal{S}$ (Equation 7) and examining its accuracy on target examples $\mathcal{T}$.

Step 1 (Pretraining): During the pre-training phase, we use inputs sampled from P_U to learn an encoder $\widehat{\phi}$ which minimizes the spectral contrastive loss (Equation 6). We define the *embedding* matrix $\widehat{F} \in \mathbb{R}^{8 \times k}$ as follows: the i-th row of $\widehat{F}$ is the feature vector for example i—that is, $\widehat{F}_i = \widehat{\phi}(i)$ where $\widehat{F}_i \in \mathbb{R}^k$ denotes the i-th row of $\widehat{F}$.

In this step of the proof we will compute $\widehat{F}$ analytically. $\widehat{F}$ is given by the top k eigenvectors and eigenvalues of the adjacency matrix A, following the analysis in HaoChen et al. (2021). We begin by computing these eigenvectors and eigenvalues.

Let A be the adjacency matrix, where $A_{ij} = S_+(i,j)$ is the probability of sampling a positive pair (i,j). Writing A out explicitly,

$$A = \frac{1}{C}\begin{bmatrix} \rho & \beta & \alpha & \gamma & 0 & 0 & \alpha & \gamma \\ \beta & \rho & \gamma & \alpha & 0 & 0 & \gamma & \alpha \\ \alpha & \gamma & \rho & \beta & \alpha & \gamma & 0 & 0 \\ \gamma & \alpha & \beta & \rho & \gamma & \alpha & 0 & 0 \\ 0 & 0 & \alpha & \gamma & \rho & \beta & \alpha & \gamma \\ 0 & 0 & \gamma & \alpha & \beta & \rho & \gamma & \alpha \\ \alpha & \gamma & 0 & 0 & \alpha & \gamma & \rho & \beta \\ \gamma & \alpha & 0 & 0 & \gamma & \alpha & \beta & \rho \end{bmatrix} \tag{75}$$

for normalization constant $C = 8\rho + 16\alpha + 8\beta + 16\gamma$.

Let $\lambda' \in \mathbb{R}^8$ be the eigenvalues of A, and $U \in \mathbb{R}^{8\times 8}$ be a matrix where the columns are corresponding unit-norm eigenvectors of A. We can explicitly write out these eigenvectors and eigenvalues as follows:

$$U = \begin{bmatrix} 1 & 1 & 0 & -1 & 0 & 1 & -1 & -1 \\ 1 & -1 & 0 & -1 & 0 & -1 & -1 & 1 \\ 1 & 0 & 1 & 0 & -1 & -1 & 1 & -1 \\ 1 & 0 & -1 & 0 & -1 & 1 & 1 & 1 \\ 1 & -1 & 0 & 1 & 0 & 1 & -1 & -1 \\ 1 & 1 & 0 & 1 & 0 & -1 & -1 & 1 \\ 1 & 0 & -1 & 0 & 1 & -1 & 1 & -1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot D, \tag{76}$$

where each column of U is an eigenvector of A, and D is a diagonal matrix that normalizes the eigenvectors to unit norm, given by:

$$D = \text{diag}\left(\left[\frac{1}{\sqrt{8}}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{\sqrt{8}}, \frac{1}{\sqrt{8}}, \frac{1}{\sqrt{8}}\right]\right). \tag{77}$$

The corresponding eigenvalues are given by $\lambda' \in \mathbb{R}^8$, where λ'_i is the eigenvalue corresponding to the i-th column of U. For convenience, let $\lambda = C\lambda'$ to avoid re-writing the C term, and we have:

$$\begin{aligned} \lambda_1 &= \rho + 2\alpha + \beta + 2\gamma \\ \lambda_2 &= \lambda_3 = -\beta + \rho \\ \lambda_4 &= \lambda_5 = \beta + \rho \\ \lambda_6 &= -2\alpha - \beta + 2\gamma + \rho \\ \lambda_7 &= -2\alpha + \beta - 2\gamma + \rho \\ \lambda_8 &= 2\alpha - \beta - 2\gamma + \rho. \end{aligned}$$

We choose $k = 3$ for the representation dimension of contrastive learning, following Theorem 1. We now show that the top 3 eigenvalues includes λ_1 and λ_8 and *does not* include λ_2 and λ_6.

Since $\rho, \alpha, \beta, \gamma > 0$, from basic algebra:

$$\lambda_1 > \lambda_4 = \lambda_5 > \lambda_2 = \lambda_3. \tag{78}$$

Since $\alpha > \gamma + \beta$ and $\beta > 0$, we also have $\alpha > \gamma$, which means:

$$\lambda_3 > \lambda_6. \tag{79}$$

Using these facts, we also get:

$$\lambda_1 > \lambda_8 > \lambda_2 = \lambda_3. \tag{80}$$

We assumed that $\alpha > \gamma + \beta$, which gives us:

$$\lambda_8 > \lambda_4 = \lambda_5. \tag{81}$$

Finally, we note that $\lambda_1 > \lambda_8$ and $\lambda_1 > \lambda_7$ because $\rho, \alpha, \beta, \gamma > 0$.

Collating all these inequalities, we find that λ_1 is the largest eigenvalue, and that λ_8 is larger than $\lambda_2, \lambda_3, \lambda_4, \lambda_5, \lambda_6$ (all eigenvalues except possibly λ_7, which can be larger or smaller than λ_8). This shows that the top 3 eigenvalues *includes* λ_1 and λ_8. Since λ_2 is smaller than $\lambda_1, \lambda_8, \lambda_4, \lambda_5$ and λ_6 is even smaller than λ_2, we also get that λ_2, λ_6 (and λ_3) are *not* among the top 3 eigenvalues.

We now write out the embedding matrix $\widehat{F}$ in terms of the top 3 eigenvectors and eigenvalues. From above, we know that the top 3 eigenvalues are $\lambda_a, \lambda_1, \lambda_8$, where $a \neq 2$ and $a \neq 6$. Let u_a, u_1, u_8 be the corresponding eigenvectors: the a-th, 1-st, and 8-th column of U respectively. From HaoChen et al. (2021) (specifically, Lemma 3.2 and then using Eckart–Young–Mirsky theorem), for some orthonormal (rotation) matrix $R \in \mathbb{R}^{k\times k}$, we have:

$$\widehat{F} = [\sqrt{\lambda'_8}u_8; \sqrt{\lambda'_1}u_1; \sqrt{\lambda'_a}u_a]R = \frac{1}{\sqrt{C}}[\sqrt{\lambda_8}u_8; \sqrt{\lambda_1}u_1; \sqrt{\lambda_a}u_a]R. \tag{82}$$

Let $\tau_8 = \sqrt{\lambda'_8} \cdot D_{88} = \sqrt{\lambda'_8}/\sqrt{8}$, $\tau_1 = \sqrt{\lambda'_1} \cdot D_{11} = \sqrt{\lambda'_1}/\sqrt{8}$, and $\tau_a = \sqrt{\lambda'_a} \cdot D_{aa}$. We can then write $\widehat{F}$ as follows, where $\tau_8, \tau_1, \tau_a > 0$. Focus on the values in the first column since this will be especially important in step 2 of the proof:

$$\widehat{F} = \begin{bmatrix} -\tau_8 & \tau_1 & \tau_a U_{1a} \\ \tau_8 & \tau_1 & \tau_a U_{2a} \\ -\tau_8 & \tau_1 & \tau_a U_{3a} \\ \tau_8 & \tau_1 & \tau_a U_{4a} \\ -\tau_8 & \tau_1 & \tau_a U_{5a} \\ \tau_8 & \tau_1 & \tau_a U_{6a} \\ -\tau_8 & \tau_1 & \tau_a U_{7a} \\ \tau_8 & \tau_1 & \tau_a U_{8a} \end{bmatrix} R. \tag{83}$$

Next, we show that $U_{1a} = U_{2a}$. To see this, recall that $a \notin \{1,2,6,8\}$. Inspecting U above in Equation 76, we see that for all other possible choices of a (so $a \in \{3,4,5,7\}$), $U_{1a} = U_{2a}$.

Therefore, we can write $\widehat{F}$ as (we have only changed the third column on the second row, notice now that the first two rows only differ on the first coordinate):

$$\widehat{F} = \begin{bmatrix} -\tau_8 & \tau_1 & \tau_a U_{1a} \\ \tau_8 & \tau_1 & \tau_a U_{1a} \\ -\tau_8 & \tau_1 & \tau_a U_{3a} \\ \tau_8 & \tau_1 & \tau_a U_{4a} \\ -\tau_8 & \tau_1 & \tau_a U_{5a} \\ \tau_8 & \tau_1 & \tau_a U_{6a} \\ -\tau_8 & \tau_1 & \tau_a U_{7a} \\ \tau_8 & \tau_1 & \tau_a U_{8a} \end{bmatrix} R. \tag{84}$$

Step 2 (Linear probing): We consider the target error of a classification head $\widehat{h}$ composed with the encoder $\widehat{\phi}$ learned by contrastive pre-training. The classification head minimizes the following objective (originally defined in Equation 7):

$$\mathcal{L}(B) = \mathbb{E}_{x \sim P_S}\left[\|\widehat{\phi}(x)^\top B - y_x\|_2^2\right] + \eta \|B\|_2^2 \tag{85}$$

where $B \in \mathbb{R}^k$, the regularization is nonzero ($\eta > 0$), and predictions are made by sign$(\widehat{\phi}(x)^\top B)$. The pre-trained encoder $\widehat{\phi}$ is fixed and obtained from step 1 above, and so the minimization is over B. In our case we only have two source labeled points, so the objective is minimized over these two labeled source points $x \in \{1,2\}$. Since the objective is rotationally symmetric, without loss of generality we can omit the rotation matrix R in the pretrained representation above in Equation 84. We consider the minimizer $\widehat{B}$ of the fine-tuning objective $\mathcal{L}$. Since the regularization strength η is strictly greater than 0, the j-th coordinate of the minimizer $\widehat{B}_j$ is 0 when the features for the labeled data $x \in \{1,2\}$ are identical on the j-th coordinate: $\widehat{\phi}(1)_j = \widehat{\phi}(2)_j$.

Note that $y_1 = 1$ and $y_2 = -1$, and $\widehat{\phi}(1)$ and $\widehat{\phi}(2)$ are given above in Equation 84. Solving the optimization problem in Equation 85 analytically (e.g., by taking derivatives setting to 0), we get for some $\omega > 0$:

$$\widehat{B} = [-\omega, 0, 0]. \tag{86}$$

Finally we show that this classifier (composed with the features learned in the previous step) predicts every target example correctly. Inspecting the feature matrix in Equation 84, we have for $x \in \{1,3,5,7\}$:

$$\widehat{B}^\top \widehat{\phi}(x) = \tau\omega > 0 \text{ and } y_x = 1, \tag{87}$$

and for $x \in \{2,4,6,8\}$,

$$\widehat{B}^\top \widehat{\phi}(x) = -\tau\omega < 0 \text{ and } y_x = -1. \tag{88}$$

In other words, sign$(\widehat{\phi}(x)^\top \widehat{B}) = y_x$ for all x, and so the target error is 0, as desired.

Conversion from augmentation to positive pair distribution. Above, we worked out the proof for contrastive pre-training in terms of the positive pair distribution $S_+(x',x)$. We showed that if $\alpha > \gamma + \beta$ then contrastive pre-training achieves 0 target error. We still need to show that such a setting exists—we started out by defining the augmentation probabilities, and we show that there is indeed some setting of $\rho', \alpha', \gamma', \beta'$ such that the condition $\alpha > \gamma + \beta$ holds.

Recall that the positive pair distribution is defined as

$$S_+(x',x) = \mathbb{E}_{\bar{x}\sim P_U}[\mathcal{A}(x \mid \bar{x})\mathcal{A}(x' \mid \bar{x})]. \tag{89}$$

The distribution S_+ induced by $\mathcal{A}$ on $\mathcal{X}$ is

$$S_+(x',x) = \begin{cases} \rho = \frac{1}{8C}[{\rho'}^2 + 2{\alpha'}^2 + {\beta'}^2 + 2{\gamma'}^2] & x = x' \\ \alpha = \frac{1}{4C}[\rho'\alpha' + \beta'\gamma'] & y_x = y_{x'}, x \neq x' \\ \beta = \frac{1}{4C}[\rho'\beta' + 2\alpha'\gamma'] & \{x',x\} \in \{\{1,2\},\{3,4\},\{5,6\},\{7,8\}\} \\ \gamma = \frac{1}{4C}[\rho'\gamma' + \alpha'\beta'] & \{x',x\} \in \{\{1,4\},\{2,3\},\{3,6\},\{4,5\},\{5,8\},\{6,7\},\{1,8\},\{2,7\}\} \end{cases} \tag{90}$$

where the constant C ensures that the probabilities sum to 1.

By setting $\gamma' = 0, \beta' = 0, \alpha' > 0, \rho' > 0$ such that $2\alpha' + \rho' = 1$, we get the desired condition $\alpha > \gamma + \beta$. This shows that there exists one such scenario, where the condition holds, as desired. However, we note that there are many situations where the condition $\alpha > \gamma + \beta$ holds.

C. Additional Details for Section 5

C.1. Additional Experiments

Table 4. Table 4 contains target test accuracies on all individual pairs of DomainNet domains (Table 2 contains only the average) with all baselines and SwAV, in addition to MoCo-V2 and MoCo-V3 (additional contrastive pre-training methods). The MoCo-V2 and MoCo-V3 hyperparameters were tuned on ImageNet in their original papers, and we did not tune either method further; therefore, for MoCo-V2 the target test accuracies are lower than several of the baselines on DomainNet (which is very different from ImageNet). However, MoCo-V3 outperforms all methods except for SwAV and DANN+strong-augs (41.59% vs. the next best, DANN: 38.38%).

Tables 5 and 6. Tables 5 and 6 contain target test accuracies on DomainNet and BREEDS of baselines with early stopping. The top halves of both tables contain results of early stopping with source test accuracy (i.e., selecting the epoch to use based on the highest source test accuracy achieved over all epochs), and the bottom halves contain results of early stopping with target test accuracy (i.e., the highest target test accuracy achieved over all epochs; this should be interpreted as an "oracle" method). As expected, early stopping improves over simply using the final iterate (as in Tables 2 and 4). However, even with early stopping only DANN+strong-augs outperforms SwAV (which uses no early stopping)—by 1.3% and 3.2% for early stopping with source and target accuracy, respectively.

Tables 7 and 8. Tables 7 and 8 contain target test accuracy when we standardize the compute requirements as much as possible between the baselines and pre-training methods, since our primary experiments allow the baselines to run for nearly twice as many epochs as the pre-training methods. Specifically, here we enforce that all methods are run for 1) the same the number of epochs with unlabeled data and 2) the same the number of epochs with unlabeled data. In particular, the baselines are initialized from an ERM baseline trained for 100 to 150 epochs, rather than 500 to 550. As a result, the resulting target accuracies are significantly lower for SENTRY and SENTRY+strong-augs (SENTRY was developed using ImageNet-pre-trained models and requires a decently accurate initialization); however, the accuracy for DANN and DANN+strong-augs is less substantial.

Table 9. To verify that other contrastive learning methods are also competitive as UDA methods, Table 9 contains target test accuracies on BREEDS of SwAV (with source-only and target-only splits of the pre-training data), MoCo-V3 (with source and target pre-training), and Dino and Barlow Twins (both with ImageNet pre-trained weights downloaded from official Github repositories). Target accuracy increases from SwAV (S) to SwAV (T) to SwAV (S+T), and the accuracies of SwAV (S+T) and MoCo (S+T) are comparable (within 1% of each other for both Living-17 and Entity-30). Dino+ and Barlow Twins+ are both higher (up to 3%) than SwAV+ on Living-17 and slightly lower (1.4%) on Entity-30.

C.2. Datasets

BREEDS. BREEDS is a subpopulation shift benchmark derived from ImageNet by constructing a hierarchical tree structure of classes from WordNet. Nodes at a specified depth of the tree become the labels for the classification task, and descendant nodes are treated as subpopulations that can be randomly partitioned into source and target domains.

We use the dataset creation functions defined in the `robustness` Python library to generate the Living-17 and Entity-30 tasks from the original ImageNet dataset (Engstrom et al., 2019; Russakovsky et al., 2015). The Living-17 dataset is an animal classification task which consists of nodes in the subtree rooted at the "living thing" node in the WordNet hierarchy. An example of a label is "ape" with subpopulations of gibbons, orangutans, gorillas, and chimpanzees. The Entity-30 dataset is a much more general task, incorporating nodes in the "entity" subtree. Labels include "building" and "home appliance". The trailing number in the dataset name is the total number of classes in that dataset.

DomainNet. DomainNet is a large unsupervised domain adaptation task, consisting of approximately 600,000 images and 345 classes in 6 domains. Each image belongs to one of 6 domains, including sketches, clipart, and photographs. For our experiments we utilize the same filtered version of DomainNet from Prabhu et al. (2021), which uses 40 of the 345 classes and the sketch, painting, photograph, and clipart domains. This refinement is done to eliminate much of the noise present in the original DomainNet dataset (Tan et al., 2020).

For our baseline experiments with the SENTRY algorithm (Prabhu et al., 2021), we use the authors' official repository (`https://github.com/virajprabhu/SENTRY`), which filters the original DomainNet dataset automatically as described above.

STL→CIFAR. CIFAR10 consists of 32×32 images from the former TinyImages dataset, and STL10 is derived from ImageNet and contains images with resolution 96×96. Both are classical image recognition datasets and are often paired together for domain shift tasks (Shu et al., 2018; French et al., 2018). We resize the STL-10 images to 32×32 to match the resolution of CIFAR10, as done in French et al. (2018). The two non-overlapping classes ("monkey" in CIFAR-10 and "frog" in STL10) are removed from both datasets before training, making the task a 9-class classification problem.

C.3. Algorithms and Hyperparameter Tuning

For joint-training baselines, we use the final iterate and we select hyperparameters based on **target test** accuracy (i.e., OOD accuracy).

ERM.

- **DomainNet.** The SENTRY algorithm runs ERM with class balancing (starting with ImageNet-pre-trained initialization) prior to beginning entropy minimization, and therefore the SENTRY repository contains an ERM implementation and hyperparameters for DomainNet. Therefore, using that ERM implementation we run ERM for 550 epochs and multiply the initial learning rate by 10x, keeping all other hyperparameters from (Prabhu et al., 2021) constant.
- **BREEDS.** We use almost the same hyperparameters as Santurkar et al. (2020). However, on Entity30, we train for 500 epochs and divide the learning rate by 10 every 500/3 = 167 epochs. On Living17, we train for 500 epochs and divide the learning rate by 10 every 167 epochs. For both datasets we use a learning rate of 0.1, a weight decay of 10^{-4}, and a batch size of 128.
- **STL → CIFAR.** The STL10 training set is much smaller than those of DomainNet and BREEDS, which gives us more freedom to sweep over hyperparameters. In turn, we vary the number of training epochs amongst $\{100,150,200,250,300\}$, the learning rate amongst $\{0.0001,0.001,0.01,0.1\}$, the softmax temperature amongst $\{0.5,0.75,1.0\}$, and the weight decay parameter amongst $\{0.0005,0.005,0.05,0.5\}$. We train the ERM model for 1000 epochs.

SENTRY. We use the official implementation (`https://github.com/virajprabhu/SENTRY`).

- **DomainNet.** For each pair of domains, we conduct a hyperparameter search through $\lambda_{\text{src}} \in \{0.5,1.0,1.5\}$ (the weight on the supervised classification loss) and learning rates $\in \{0.01,0.001\}$. We run all hyperparameter settings for 100 epochs from a 150-epoch ERM checkpoint and select the best one (based on target test accuracy). We then initialize the SENTRY model with the ERM model described earlier (i.e., trained for 550 epochs) and run SENTRY for 400 epochs (for a total of 950 epochs).

- **BREEDS.** We keep the default hyperparameters from the SENTRY repo but search over 3 learning rates $\{0.001, 0.01, 0.1\}$ for 100 epochs and then run the best hyperparameter setting for 300 additional epochs for 400 total epochs. We initialize the model with the ERM model described earlier (i.e., trained for 500 epochs).
- **STL → CIFAR.** We initialize SENTRY with an ERM model trained for 100 epochs and then train SENTRY for 1000 epochs, varying the learning rate between $\{0.001, 0.01, 0.1\}$, the weight decay between $\{0.0005, 0.005, 0.05, 0.5\}$, and the unsupervised loss weight between $\{0.01, 0.1, 1.0\}$.

For the "strong augmentation" version of SENTRY, we replace the RandAug algorithm used for consistency regularization with the augmentations used for SwAV (in DomainNet and BREEDS) or SimCLR (for STL / CIFAR).

DANN. We use the implementation provided in the SENTRY repository (https://github.com/virajprabhu/SENTRY).

- **DomainNet.** For each pair of domains, we conduct a hyperparameter search through learning rates $\in \{0.01, 0.001\}$ and temperature $\in \{0.9, 1.0\}$. As with SENTRY, we run all hyperparameter setting for 100 epochs from a 150-epoch ERM checkpoint, select the best one (based on target test accuracy). We initialize the DANN model with the ERM model described earlier (i.e., trained for 550 epochs) and run DANN for 400 epochs (for a total of 950 epochs).
- **BREEDS.** We sweep over two learning rates of 0.001 and 0.0005 for 100 epochs and choose the learning rate that achieves the highest OOD accuracy. We then run DANN for 300 additional epochs for 400 total epochs. We initialize the SENTRY model with the ERM model described earlier.
- **STL → CIFAR.** We follow an identical procedure to that of SENTRY, initializing DANN with an ERM model trained for 100 epochs and sweeping over the same hyperparameter set. DANN was run for 1000 epochs.

For the "strong augmentation" version of DANN we simply replace the input augmentations with the augmentations used by SwAV (for DomainNet and BREEDS) or SimCLR (for STL / CIFAR) and initialize DANN from an ERM checkpoint trained with SwAV augmentations.

DIRT-T. DIRT-T (Shu et al., 2018) is a domain adaptation method that addresses two flaws of domain adversarial neural networks (Ganin et al., 2016): 1) distribution matching is a weak constraint, and 2) in some domain adaptation settings there does not exist a good joint classifier on both source and target. The authors address the first shortcoming by adding a conditional entropy regularization term so that the model's decision boundaries do not overlap high-density regions of data. This is inspired by the *cluster assumption*, which states that the input space is divided into well-separated clusters, one for each class in the label space. The lack of a good classifier on both source and target is then addressed via self-training on the unlabeled target data. We report the STL→CIFAR DIRT-T results from Shu et al. (2018).

C.4. SwAV

We use the official SwAV implementation (https://github.com/facebookresearch/swav) and keep almost all of the hyperparameters provided by the original paper for 400 epoch, 256-batch-size training on ImageNet. However, we use a batch size of 512 and tuned SwAV slightly on Living-17 using only the training curves (no labels) and following the advice from the Github README and issues answered by the original authors:

- In order to stabilize training, we do not use a queue on DomainNet and the subsampled variants of Living-17. For pre-training on the full Living-17 and Entity-30 datasets, we introduce the queue at epoch 60.
- We set the number of prototypes to be 10 times the number of classes (170, 300, and 400 for Living-17, Entity-30 and DomainNet, respectively).
- We set $\epsilon = 0.03$.
- We set the base learning rate to 0.6, following a linear scaling rule based on batch size.

We note that the joint-training baselines required extensive hyperparamter tuning for *every* transfer task. In contrast, because SwAV was tuned for ImageNet training, we selected hyperparameters once (using only the pre-training loss and no labels on Living-17) and use the same hyperparameters (except for the queue) on all our datasets. For fine-tuning, we always initialize with the final iterate of SwAV pre-training (400 epochs). Additional dataset-specific details:

- **DomainNet.** For consistency, we fine-tune SwAV pre-trained models for 150 epochs with strong data augmentations using the ERM implementation in the SENTRY repository (without running any joint-training algorithm), keeping all hyperparameters other than the number of epochs constant. We report the target test accuracy of the final iterate (i.e., after 150 epochs).
- **Living-17.** We fine-tune SwAV models for 100 epochs strong data augmentations and with a cosine learning rate schedule without restarts. We use SGD with initial learning rate 0.1 for the classifier head and 0.01 for the encoder, momentum 0.9, and weight decay 0.0001. We use a batch size of 96, and once again report the target test accuracy of the final model (i.e., after 100 epochs).
- **Entity-30.** We linear probe for 100 epochs instead of fine-tune on Entity-30, due to its large size (300k examples).

C.5. SimCLR

We use the official SimCLR implementation (`https://github.com/google-research/simclr`). We use a batch size of 256, a learning rate of 0.2, and weight decay 10^{-4}. The projection head has two layers and an output dimension of 64. We pre-train the model for 1000 epochs with square-root learning rate scaling and we train the linear probe for 100 epochs on batches of size 512 and a learning rate of 0.1.

C.6. MoCo-V2 and MoCo-V3

We use the official MoCo implementations (`https://github.com/facebookresearch/moco` and `https://github.com/facebookresearch/moco-v3`). We use a batch size of 512 and otherwise all the default hyperparameters for MoCo-V2. We use a batch size of 256 and otherwise all the defualt hyperparameters for MoCo-V3. We fine-tune both MoCo versions on DomainNet and Living-17 using the same protocol as SwAV fine-tuning for each dataset.

C.7. Dino and Barlow Twins

We use ImageNet pre-trained weights from the official Github repositories (`https://github.com/facebookresearch/dino` and `https://github.com/facebookresearch/barlowtwins`). We fine-tune on Living-17 and linear probe on Entity-30 using the same protocol as for SwAV. We note that while SwAV+extra is pre-trained for 400 epochs, the Dino and Barlow Twins models were pre-trained for 800 and 1000 epochs, respectively.

C.8. Model architectures.

For all BREEDS experiments, we use a standard ResNet50 from the PyTorch `torchvision` library. For DomainNet, we use a ResNet50 slightly modified for few-shot learning, following Prabhu et al. (2021). On STL→CIFAR for ERM, DANN, SENTRY, and SimCLR baselines we use a standard ResNet18 from `torchvision`, and we additionally report the DIRT-T performance from Shu et al. (2018), which uses a custom 18-layer CNN.

D. Additional Experiments Verifying the Theory

D.1. Target accuracy as a function of connectivity

Eq. 8 can be rewritten as

$$\log(\text{target accuracy}) \approx w_1 \cdot \log(\alpha/\gamma) + w_2 \cdot \log(\beta/\gamma) \tag{91}$$

and for each baseline, we fit w_1 and w_2 using a linear regression model on data from the 12 source/target domain pairs of DomainNet. We then compute the coefficient of determination R^2 between the observed and predicted target accuracies. The fitted exponents w_1,w_2 and R^2 for SwAV, MoCo-V2, and MoCo-V3 are provided in Table 10. Figures 5 and 6 plot the observed target accuracies of SwAV, MoCo-V2, MoCo-V3, DANN, and SENTRY (y-axis) against the predicted (x-axis), along with a line of best fit through them as a visual aid. We find that R^2 is high for contrastive learning (0.78, 0.79, and 0.60) but relatively low for the baselines ($R^2 \in [0.23, 0.51]$ for DANN and $R^2 \in [-0.20, 0.22]$ for SENTRY).

D.2. Ablation of connectivity

For this experiment on Living-17, we used a modified version of the dataset as follows: for each of the 17 classes, we trained a ResNet-50 classifier to distinguish between augmented images of that class in the source and target. Of the resulting classifiers, we selected the 4 that obtained domain classification accuracy $> 70\%$: these were classes 6, 11, 12, and 14. We only kept those 4 classes, and all data selection methods we considered chose subsets from this modified dataset.

D.3. Disentanglement of class and domain information

Table 11 contains individual results on the cosine similarity between domain and class classifiers on DomainNet (abbreviated version in Table 3). In the fine-tuned feature space, the class and domain classifiers remain orthogonal while the cosine similarity between source and target classifiers increases by 41% from 0.187 to 0.264.

E. Results and Protocol for Estimating Connectivity Parameters on Benchmark Datasets

Table 12 reports the connectivity estimates for the input space and the feature spaces learned by SwAV and DANN+strong-augs for all pairs of DomainNet domains (abbreviated version in Table 1). To estimate the average connectivity between two class-domain pairs (c,d) and (c',d'), we use the following algorithm:

1. Label all training examples of class c and domain d as 0 and all training examples of class c' and domain d' as 1. Discard the remaining examples from other classes/domains.
2. Train a ResNet50 for 100 epochs using strong augmentations and a cosine learning rate. We did not exhaustively tune this training step, but we kept the procedure constant for all classes and domains. If estimating connectivity in the input space, we train the entire network; if estimating in the feature space, we train only a linear classifier with a frozen encoder.
3. Create the test set analogously to step 1, and evaluate the classifier on strongly augmented data. The test error of the classifier is interpreted as an estimate for connectivity between the two class-domain pairs.

Each domain in DomainNet has a unique label distribution (all of which are far from uniform), and therefore in computing the average connectivity we compute the weighted mean, where each pair of (class, domain) pairs is weighted by the ratio of the less to more frequent label (0 or 1).

In the SwAV feature space, the across-domain and across-class connectivities are approximately equal (7.54 vs. 7.03). On other hand, the across-domain is much higher than across-class in the DANN+strong-augs feature space (13.64 vs. 5.65); intuitively, the classification component of the DANN objective pushes apart the classes in feature space, while the domain discrimination component brings together domains. The connectivities (i.e., classifier error) are all much higher in the input space because the classifiers are trained from scratch and on much smaller datasets, while for SwAV and DANN the classifiers are trained via linear probing in the feature space. Thus, to standardize compute between the methods, we used the DANN+strong-augs checkpoints from the compute-standardized experiment (discussed in Section C.1 and Tables 7 and 8).

F. Appendix Figures and Tables

The tables and figures for the appendix are listed below in order to improve readability in the previous sections.

Source	Real			Sketch			Painting			Clipart			Avg.
Target	Sketch	Painting	Clipart	Real	Painting	Clipart	Real	Sketch	Clipart	Real	Sketch	Painting	
ERM	29.55	43.25	45.92	28.09	20.66	34.41	33.86	16.63	22.83	20.68	12.59	11.52	26.67
ERM (+DA)	42.31	48.47	49.75	36.47	34.65	44.68	45.59	39.47	35.33	27.93	31.18	16.36	37.68
SENTRY	42.89	57.03	**65.41**	**55.81**	**42.97**	53.03	42.98	1.92	2.10	8.84	4.42	1.58	31.58
SENTRY (+DA)	40.43	48.92	54.02	44.00	29.39	51.18	3.51	2.58	23.82	3.60	2.88	5.50	25.82
DANN	43.93	49.12	56.06	44.46	36.06	48.33	44.29	31.89	30.01	33.30	25.59	17.46	38.38
DANN (+DA)	**54.19**	51.29	58.66	48.22	41.46	**54.95**	52.40	**49.06**	**37.75**	38.01	**41.52**	22.24	**45.81**
MoCo-V2	34.88	43.79	47.77	36.07	23.85	32.92	47.86	29.38	25.43	34.55	23.75	14.91	32.93
MoCo-V3	44.60	**57.82**	53.53	48.85	29.84	32.67	63.86	36.47	31.99	50.01	27.97	21.45	41.59
SwAV	43.76	54.55	53.27	55.48	34.99	40.59	**67.08**	39.64	32.98	**57.13**	34.30	**25.09**	44.91
SwAV+	44.64	57.27	54.20	58.10	46.75	53.46	69.03	48.68	41.33	59.38	46.22	41.66	51.73

Table 4: Test accuracy (%) of baselines, MoCo-V2, MoCo-V3, SwAV, and SwAV+extra on all individual domain pairs of DomainNet. SwAV on average is within 1% of the best baseline, DANN+strong-augs (44.91 vs. 45.81).

Source	Real			Sketch			Painting			Clipart			Avg.
Target	Sketch	Painting	Clipart	Real	Painting	Clipart	Real	Sketch	Clipart	Real	Sketch	Painting	
Early stopping using source test accuracy													
SENTRY	43.27	55.00	63.49	54.96	42.87	51.67	41.83	20.22	25.43	21.23	10.38	10.42	36.73
SENTRY (+DA)	36.18	49.54	59.96	51.33	29.98	54.39	33.88	10.13	20.73	18.03	15.42	9.52	32.42
DANN	40.31	46.17	53.09	48.71	38.29	51.42	46.36	31.72	28.90	34.06	27.05	17.19	38.61
DANN (+DA)	55.86	52.01	57.49	46.35	42.35	56.81	50.64	50.15	42.20	37.72	40.39	23.00	46.25
Early stopping using target test accuracy													
SENTRY	48.77	58.74	66.15	58.74	43.62	54.14	47.40	20.25	28.15	22.10	10.71	11.17	39.16
SENTRY (+DA)	42.35	53.66	63.67	51.33	31.90	58.47	34.36	16.13	27.97	18.66	15.42	9.52	35.29
DANN	45.18	50.01	56.93	50.03	40.73	52.84	47.11	34.76	32.17	35.82	28.01	20.07	41.14
DANN (+DA)	56.69	53.76	61.50	49.44	42.38	58.10	52.39	50.93	43.37	39.08	44.85	24.95	48.12

Table 5: Test accuracy (%) of DANN and SENTRY on all individual domain pairs of DomainNet with early stopping using source test accuracy (top) and target test accuracy (bottom) as opposed to using the final iterate, which is used in Table 2. Early stopping SENTRY consistently leads to large boosts in accuracy (5.1% and 7.5% for early stopping with source and target), while early stopping DANN leads to much smaller boosts (0.2% and 2.7% for early stopping with source and target). Compared to SwAV, with either method of early stopping, SENTRY and SENTRY+strong-augs are more than 5% lower and DANN is more than 3% lower. However, DANN+strong-augs is better than SwAV by 1.3% and 3.2% for early stopping with source and target accuracy, respectively.

	Living-17	Entity-30
Early stopping using source test accuracy		
SENTRY	79.24	63.45
SENTRY (+DA)	76.06	61.55
DANN	67.41	53.30
DANN (+DA)	70.12	53.68
Early stopping using target test accuracy		
SENTRY	79.94	64.07
SENTRY (+DA)	77.18	64.10
DANN	69.59	59.10
DANN (+DA)	73.18	57.82

Table 6: Test accuracy (%) of DANN and SENTRY on Living-17 and Entity-30 with early stopping using source test accuracy (top) and target test accuracy (bottom) as opposed to using the final iterate, which is used in Table 2. Both methods of early stopping SENTRY and SENTRY+strong-augs led to boosts in accuracy over using the final iterate, but early stopping DANN and DANN+strong-augs with source accuracy inconsistently increased performance over the final iterate.

Source	Real			Sketch			Painting			Clipart			Avg.
Target	Sketch	Painting	Clipart	Real	Painting	Clipart	Real	Sketch	Clipart	Real	Sketch	Painting	
SENTRY	39.39	48.33	55.63	56.00	39.22	48.02	17.77	2.13	3.90	3.79	2.33	1.44	26.50
SENTRY (+DA)	37.14	36.58	51.42	44.88	23.20	46.23	14.35	7.50	12.93	3.27	7.00	2.65	23.93
DANN	44.02	43.86	51.18	44.42	37.47	52.10	47.96	31.10	32.43	32.36	24.26	15.30	38.04
DANN (+DA)	50.77	50.05	56.37	47.90	40.39	55.14	50.47	46.23	40.10	37.38	41.60	23.55	45.00

Table 7: Test accuracy (%) of DANN and SENTRY on all pairs of DomainNet with standardized compute resources as described in Section C.1. On average, DANN and DANN+strong-augs achieve comparable results to those in the main table (Table 2), while SENTRY and SENTRY+strong-augs are both lower here than in the main table.

	Living-17	Entity-30
SENTRY	60.65	54.71
SENTRY (+DA)	69.12	56.55
DANN	65.65	55.93
DANN (+DA)	68.06	56.00

Table 8: Test accuracy (%) of DANN and SENTRY on Living-17 and Entity-30 with standardized compute resources as described in Section C.1. For all 4 methods, the accuracies shown here are somewhat close to ($\leq 2.5\%$ lower than) those shown in the main table (Table 2).

	SwAV (S)	SwAV (T)	SwAV (S+T)	MoCo-V3 (S+T)	SwAV+	Dino+	Barlow Twins+
Living-17	62.71	70.41	75.12	74.88	82.00	83.05	85.11
Entity-30	52.33	60.33	62.03	63.00	65.90	64.50	64.48

Table 9: Test accuracy (%) of additional contrastive pre-training methods on Living-17 and Entity-30: SwAV (with source-only and target-only pre-training), Moco-V3 (source and target pre-training), Dino+ (ImageNet pre-training), and Barlow Twins+ (ImageNet pre-training). For ease of comparison, SwAV (S+T) and SwAV+ results are repeated from Table 2. On both datasets, with source and target pre-training MoCo-V3 is comparable with SwAV and with ImageNet pre-training Dino and Barlow Twins are comparable with SwAV.

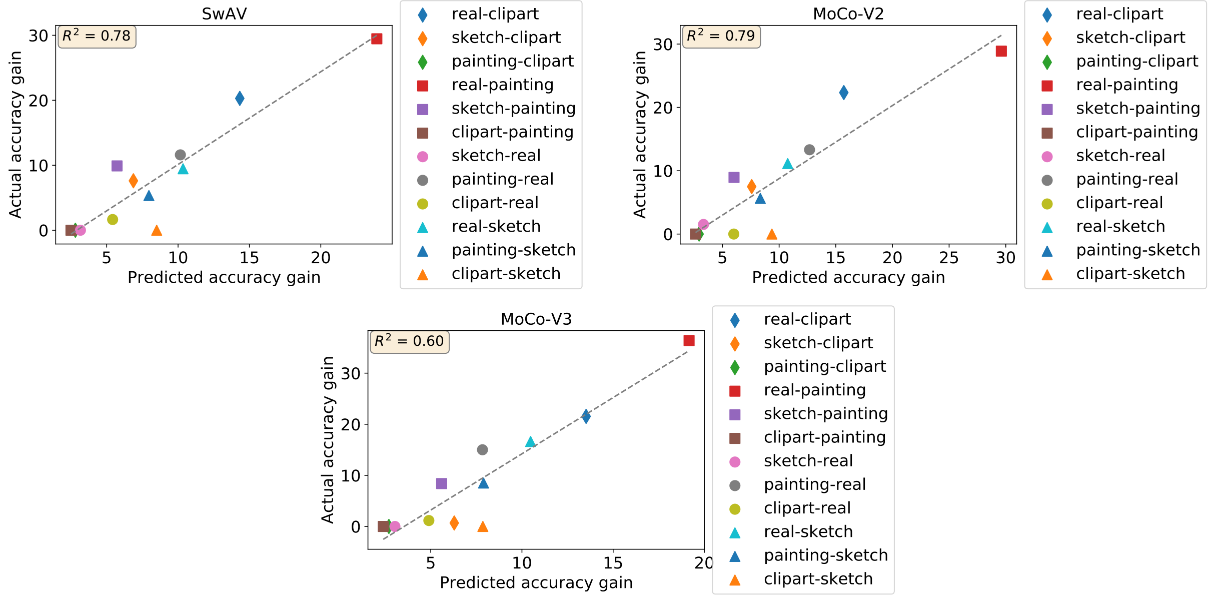

Figure 5: Target accuracies observed vs. accuracies predicted using a function of the connectivity ratios. The coefficient of determination (R^2) is reported in the upper left corner of each plot. The top left panel reports SwAV results (repeated from the left panel of Fig. 3 for ease of comparison), the top right panel reports MoCo-V2 results, and the bottom panel reports MoCo-V3 results. The line of best fit between the observed and predicted is also plotted to help visualize the relationship.

	Across-domain / across-both coefficient (w_1)	Across-class / across-both coefficient (w_2)	Coefficient of determination (R^2)
SwAV	14.86	2.67	0.78
MoCo	16.07	2.65	0.79
MoCo-V3	13.51	2.79	0.60

Table 10: Values of w_1 and w_2 fitted to Eq. 91, along with the coefficients of determination (R^2) between the observed and predicted target accuracies. Accuracies of contrastive pre-training methods (SwAV, MoCo-V2, and MoCo-V3) are well-explained using the connectivity ratios (R^2 of 0.78, 0.79, 0.60, respectively).

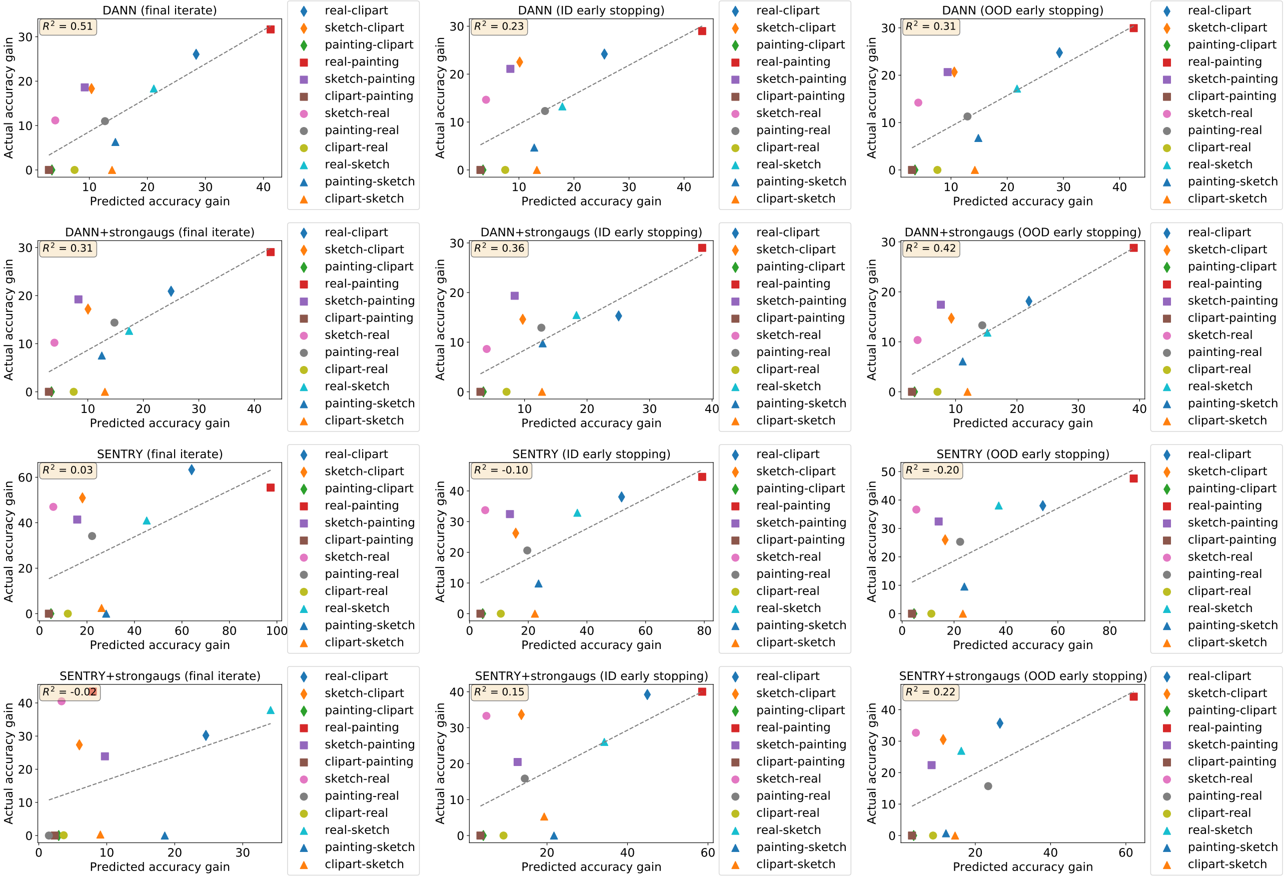

Figure 6: Target accuracies observed vs. accuracies predicted using a function of the connectivity ratios. The coefficient of determination (R^2) is reported in the upper left corner of each plot. The 4 rows contain results of DANN, DANN+strong-augs, SENTRY, and SENTRY+strong-augs(in that order). The 3 columns contain results from different methods of early stopping: final iterate (i.e., no early stopping), source test, and target test (in that order). The connectivity ratio is a somewhat poor predictor of the observed accuracies for DANN and DANN+strong-augs ($R^2 \in [0.23, 0.51]$) and an extremely poor predictor for SENTRY and SENTRY+strong-augs ($R^2 \in [-0.20, 0.22]$). Note that because we do not fit an intercept, the computation of the coefficient of determination may produce a negative number. The line of best fit between the observed and predicted is also plotted to help visualize the relationship.

	Source vs. Target		Source vs. Domain		Target vs. Domain	
	Pre-trained	Fine-tuned	Pre-trained	Fine-tuned	Pre-trained	Fine-tuned
Real → Sketch	0.200	0.286	0.016	0.011	0.019	0.020
Real → Painting	0.228	0.305	0.017	0.014	0.019	0.021
Real → Clipart	0.222	0.319	0.016	0.013	0.017	0.018
Sketch → Real	0.200	0.263	0.019	0.019	0.016	0.016
Sketch → Painting	0.176	0.254	0.015	0.016	0.021	0.021
Sketch → Clipart	0.159	0.250	0.018	0.011	0.019	0.019
Painting → Real	0.228	0.271	0.019	0.017	0.017	0.014
Painting → Sketch	0.176	0.240	0.021	0.022	0.015	0.018
Painting → Clipart	0.138	0.233	0.019	0.016	0.024	0.020
Clipart → Real	0.222	0.277	0.017	0.018	0.016	0.010
Clipart → Sketch	0.159	0.234	0.019	0.021	0.018	0.013
Clipart → Painting	0.138	0.237	0.024	0.022	0.019	0.016
Average (DomainNet)	0.187	**0.264**	**0.018**	0.017	**0.018**	0.017
% change (Fine-tune vs. Pre-train)		**+41%**		**-5.5%**		**-5.5%**

Table 11: Cosine similarity of class and domain classifiers trained on SwAV representations (average over all classes) on DomainNet. Class classifiers trained on the source and target individually learn similar linear weights, evidenced by the average cosine similarity 0.18. However, domain classifiers learn linear weights that are nearly orthogonal to the class classifier weights, suggesting that SwAV pre-training learns features containing domain and class information in somewhat separate directions.

Input space	Different class, same domain, domain 1	Different class, same domain, domain 2	Same class, different domain	Different class, different domain
Real ↔ Sketch	24.49	38.12	21.47	20.51
Real ↔ Painting	24.49	33.71	33.62	28.07
Real ↔ Clipart	24.49	35.20	23.12	21.18
Sketch ↔ Painting	38.12	33.71	24.34	23.16
Sketch ↔ Clipart	38.12	35.20	30.23	27.72
Painting ↔ Clipart	33.71	35.20	31.36	30.09
Avg.	32.88		27.36	25.12

SwAV pre-trained space	Different class, same domain, domain 1	Different class, same domain, domain 2	Same class, different domain	Different class, different domain
Real ↔ Sketch	3.08	6.92	4.73	1.74
Real ↔ Painting	3.07	7.43	11.79	2.23
Real ↔ Clipart	3.02	6.34	8.93	1.73
Sketch ↔ Painting	8.50	10.65	5.55	2.66
Sketch ↔ Clipart	8.54	8.15	8.10	3.29
Painting ↔ Clipart	10.22	8.40	6.13	3.12
Avg.	7.03		7.54	2.46

DANN+strong-augs feature space	Different class, same domain, source	Different class, same domain, target	Same class, different domain	Different class, different domain
Real → Sketch	2.72	5.70	7.72	2.28
Real → Painting	2.54	7.44	16.85	3.36
Real → Clipart	2.91	5.45	10.45	1.92
Sketch → Real	4.54	4.72	13.32	4.30
Sketch → Painting	4.69	9.79	14.27	5.12
Sketch → Clipart	4.85	5.49	17.44	4.64
Painting → Real	7.29	3.76	20.71	4.87
Painting → Sketch	7.43	6.54	12.45	4.57
Painting → Clipart	7.90	5.90	10.76	3.63
Clipart → Real	4.42	5.27	14.38	3.13
Clipart → Sketch	6.21	3.93	13.84	4.50
Clipart → Painting	4.42	11.69	11.52	4.32
Avg.	4.99	6.31	13.64	3.89

Table 12: Empirical estimates of the different parameters of connectivity in the input space (top) and feature spaces computed by SwAV (middle) and DANN+strong-augs (bottom). The numbers provided are the error of classifying different types of class-domain pairs.