

# BREEDS: Benchmarks for Subpopulation Shift

Shibani Santurkar\*  
 MIT  
 shibani@mit.edu

Dimitris Tsipras\*  
 MIT  
 tsipras@mit.edu

Aleksander Mądry  
 MIT  
 madry@mit.edu

## Abstract

We develop a methodology for assessing the robustness of models to subpopulation shift—specifically, their ability to generalize to novel data subpopulations that were not observed during training. Our approach leverages the class structure underlying existing datasets to control the data subpopulations that comprise the training and test distributions. This enables us to synthesize realistic distribution shifts whose sources can be precisely controlled and characterized, within existing large-scale datasets. Applying this methodology to the ImageNet dataset, we create a suite of subpopulation shift benchmarks of varying granularity. We then validate that the corresponding shifts are tractable by obtaining human baselines for them. Finally, we utilize these benchmarks to measure the sensitivity of standard model architectures as well as the effectiveness of off-the-shelf train-time robustness interventions.<sup>1</sup>

## 1 Introduction

Robustness to distribution shift has been the focus of a long line of work in machine learning [SG86; WK93; KHA99; Shi00; SKM07; Qui+09; Mor+12; SK12]. At a high-level, the goal is to ensure that models perform well not only on unseen samples from the datasets they are trained on, but also on the diverse set of inputs they are likely to encounter in the real world. However, building benchmarks for evaluating such robustness is challenging—it requires modeling realistic data variations in a way that is well-defined, controllable, and easy to simulate.

Prior work in this context has focused on building benchmarks that capture distribution shifts caused by natural or adversarial input corruptions [Sze+14; FF15; FMF16; Eng+19a; For+19; HD19; Kan+19], differences in data sources [Sae+10; TE11; Kho+12; TT14; Rec+19], and changes in the frequencies of data subpopulations [Ore+19; Sag+20]. While each of these approaches captures a different source of real-world distribution shift, we cannot expect any single benchmark to be comprehensive. Thus, to obtain a holistic understanding of model robustness, we need to keep expanding our testbed to encompass more natural modes of variation. In this work, we take another step in that direction by studying the following question:

*How well do models generalize to data subpopulations they have not seen during training?*

The notion of *subpopulation shift* this question refers to is quite pervasive. After all, our training datasets will inevitably fail to perfectly capture the diversity of the real word. Hence, during deployment, our models are bound to encounter unseen subpopulations—for instance, unexpected weather conditions in the self-driving car context or different diagnostic setups in medical applications.

## Our contributions

The goal of our work is to create large-scale subpopulation shift benchmarks wherein the data subpopulations present during model training and evaluation differ. These benchmarks aim to assess how effectively models generalize beyond the limited diversity of their training datasets—e.g., whether models can recognize

---

\*Equal contribution.

<sup>1</sup>Code and data available at <https://github.com/MadryLab/BREEDS-Benchmarks>.

Dalmatians as “dogs” even when their training data for “dogs” comprises only Poodles and Terriers. We show how one can simulate such shifts, fairly naturally, *within* existing datasets, hence eliminating the need for (and the potential biases introduced by) crafting synthetic transformations or collecting additional data.

**BREEDS benchmarks.** The crux of our approach is to leverage existing dataset labels and use them to identify *superclasses*—i.e., groups of semantically similar classes. This allows us to construct classification tasks over such superclasses, and repurpose the original dataset classes to be the subpopulations of interest. This, in turn, enables us to induce a subpopulation shift by directly making the subpopulations present in the training and test distributions disjoint. By applying this methodology to the ImageNet dataset [Den+09], we create a suite of subpopulation shift benchmarks of varying difficulty. This involves modifying the existing ImageNet class hierarchy—WordNet [Mil95]—to ensure that superclasses comprise visually coherent subpopulations. We then conduct human studies to validate that the resulting BREEDS benchmarks indeed capture meaningful subpopulation shifts.

**Model robustness to subpopulation shift.** In order to demonstrate the utility of our benchmarks, we employ them to evaluate the robustness of standard models to subpopulation shift. In general, we find that model performance drops significantly on the shifted distribution—even when this shift does not significantly affect humans. Still, models that are more accurate on the original distribution tend to also be more robust to these subpopulation shifts. Moreover, adapting models to the shifted domain, by retraining their last layer on data from this domain, only partially recovers the original model performance.

**Impact of robustness interventions.** Finally, we examine whether various train-time interventions, designed to decrease model sensitivity to synthetic data corruptions (e.g.,  $\ell_2$ -bounded perturbations) make models more robust to subpopulation shift. We find that many of these methods offer small, yet non-trivial, improvements to model robustness along this axis—at times, at the expense of performance on the original distribution. Often, these improvements become more pronounced after retraining the last layer of the model on the shifted distribution. In the context of adversarial training, our findings are in line with recent work showing that the resulting robust models often exhibit improved robustness to other data corruptions [For+19; Kan+19; Tao+20], and transfer better to downstream tasks [Utr+20; Sal+20]. Nonetheless, none of these interventions significantly alleviate model sensitivity to subpopulation shift, indicating that the BREEDS benchmarks pose a challenge to current methods.

## 2 Designing Benchmarks for Distribution Shift

When constructing distribution shift benchmarks, the key design choice lies in specifying the *target distribution* to be used during model evaluation. This distribution is meant to be a realistic variation of the *source distribution*, that was used for training. Typically, studies focus on variations due to:

- *Data corruptions*: The target distribution is obtained by modifying inputs from the source distribution via a family of transformations that mimic real-world corruptions. Examples include natural or adversarial forms of noise [FF15; FMF16; Eng+19a; HD19; For+19; Kan+19; Sha+19].
- *Differences in data sources*: Here, the target distribution is an independently collected dataset for the same task [Sae+10; TE11; TT14; BVP18; Rec+19]—for instance, using PASCAL VOC [Eve+10] to evaluate ImageNet-trained classifiers [Rus+15]. The goal is to test whether models are overly reliant on the idiosyncrasies of the datasets they are trained on [Pon+06; TE11].
- *Subpopulation shifts*: The source and target distributions differ in terms of how well-represented each subpopulation is. Work in this area typically studies whether models perform *equally well* across all subpopulations from the perspective of reliability [M+15; Hu+18; DN18; Cal+18; Ore+19; Sag+20] or algorithmic fairness [Dwo+12; KMR17; JTJ17; BG18; Has+18].

In general, a major challenge lies in ensuring that the distribution shift between the source and target distributions (also referred to as *domains*) is caused solely by the intended input variations. External factors—which may arise when crafting synthetic transformations or collecting new data—could skew the target

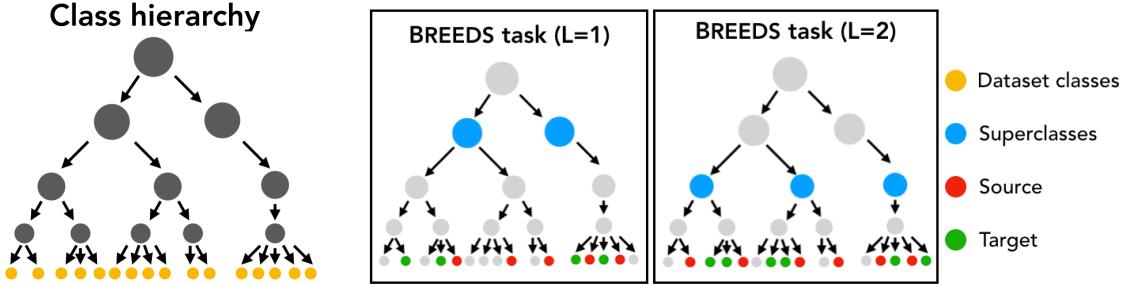


Figure 1: Illustration of our pipeline to create subpopulation shift benchmarks. Given a dataset, we first define superclasses by grouping semantically similar classes together to form a hierarchy. This allows us to treat the dataset labels as subpopulation annotations. Then, we construct a BREEDS task of specified granularity (i.e., depth in the hierarchy) by posing the classification task in terms of superclasses at that depth and then partitioning their respective subpopulations into the source and target domains.

distribution in different ways, making it hard to gauge model robustness to the exact distribution shift of interest. For instance, recent work [Eng+20] demonstrates that collecting a new dataset while aiming to match an existing one along a specific metric (e.g., as in Recht et al. [Rec+19]) might result in a miscalibrated dataset due to statistical bias. In our study, we aim to limit such external influences by simulating shifts within existing datasets, thus avoiding any input modifications.

### 3 The BREEDS Methodology

In this work, we focus on modeling a pertinent, yet relatively less studied, form of subpopulation shift: one wherein the target distribution (used for testing) contains subpopulations that are *entirely* absent from the source distribution that the model was trained on. To simulate such a shift, one needs to precisely control the data subpopulations that comprise the source and target data distributions. Our procedure for doing this comprises two stages that are outlined below—see Figure 1 for an illustration.

**Devising subpopulation structure.** Typical datasets do not contain annotations for individual subpopulations. Since collecting such annotations would be challenging, we take an alternative approach: we bootstrap the existing dataset labels to simulate subpopulations. That is, we group semantically similar classes into broader superclasses which, in turn, allows us to re-purpose existing class labels as the desired subpopulation annotations. In fact, we can group classes in a hierarchical manner, obtaining superclasses of different specificity. As we will see in Section 4, large-scale benchmarks often provide class hierarchies [Eve+10; Den+09; Kuz+18] that aid such semantic grouping.

**Simulating subpopulation shifts.** Given a set of superclasses, we can define a classification task over them: the inputs of each superclass correspond to pooling together the inputs of its subclasses (i.e., the original dataset classes). Within this setup, we can simulate subpopulation shift in a relatively straightforward manner. Specifically, for each superclass, we split its subclasses into two *random* and *disjoint* sets, and assign one of them to the source and the other to the target domain. Then, we can evaluate model robustness under subpopulation shift by simply training on the source domain and testing on the target domain. Note that the classification task remains identical between domains—both domains contain the same (super)classes but the subpopulations that comprise each (super)class differ.<sup>2</sup> Intuitively, this corresponds to using different dog breeds to represent the class “dog” during training and testing—hence the name of our toolkit.

This methodology is quite general and can be applied to a variety of setting to simulate realistic distribution shifts. Moreover, it has a number of additional benefits:

<sup>2</sup>Note that this methodology can be extended to simulate milder subpopulation shifts where the source and target distributions overlap but the relative subpopulation frequencies vary, similar to the setting of Oren et al. [Ore+19].

- **Flexibility:** Different semantic groupings of a fixed set of classes lead to BREEDS tasks of varying granularity. For instance, by only grouping together classes that are quite similar one can reduce the severity of the subpopulation shift. Alternatively, one can consider broad superclasses, each having multiple subclasses, resulting in a more challenging benchmark.
- **Precise characterization:** The exact subpopulation shift between the source and target distribution is known. Since both domains are constructed from the same dataset, the impact of any external factors (e.g., differences in data collection pipelines) is minimized. (Note that such external factors can significantly impact the difficulty of the task [Pon+06; TE11; Eng+20; Tsi+20].)
- **Symmetry:** Since subpopulations are split into the source and test domains randomly, we expect the resulting tasks to have comparable difficulty.
- **Reuse of existing datasets:** No additional data collection or annotation is required other than choosing the class grouping. This approach can thus be used to also re-purpose other existing large-scale datasets—even outside the image recognition context—with minimal effort and cost.

## 4 Simulating Subpopulation Shifts Within ImageNet

We now describe in more detail how our methodology can be applied to ImageNet [Den+09]—specifically, the ILSVRC2012 subset [Rus+15]—to create a suite of BREEDS benchmarks. ImageNet contains a large number of classes, making it particularly well-suited for our purpose.

### 4.1 Utilizing the ImageNet class hierarchy

Recall that creating BREEDS tasks requires grouping together similar classes. In the context of ImageNet, such a semantic grouping already exists—ImageNet classes are a part of the WordNet hierarchy [Mil95]. However, WordNet is not a hierarchy of objects but rather one of word meanings. Therefore, intermediate hierarchy nodes are not always well-suited for object recognition due to:

- **Abstract groupings:** WordNet nodes often correspond to abstract concepts, e.g., related to the functionality of an object. Children of such nodes might thus share little visual similarity—e.g., “umbrella” and “roof” are visually different, despite both being “coverings”.
- **Non-uniform categorization:** The granularity of object categorization is vastly different across the WordNet hierarchy—e.g., the subtree rooted at “dog” is 25-times larger than the one rooted at “cat”. Hence, the depth of a node in this hierarchy does not always reflect the specificity of the corresponding object category.
- **Lack of tree structure:** Nodes in WordNet can have multiple parents<sup>3</sup> and thus the resulting classification task would contain overlapping classes, making it inherently ambiguous.

Due to these issues, we cannot directly use WordNet to identify superclasses that correspond to a well-calibrated classification task. To illustrate this, we present some of the superclasses constructed by applying clustering algorithms directly to the WordNet hierarchy [HAE16] in Appendix Table 2. Even putting the issue of overlapping classes aside, a BREEDS task based on these superclasses would induce a very skewed subpopulation shift across classes—e.g., varying the types of “bread” is very different than doing the same for different “mammal” species.

**Calibrating WordNet for Visual Object Recognition.** To better align the WordNet hierarchy with the task of object recognition in general, and BREEDS benchmarks in particular, we manually modify it according to the following two principles. First, nodes should be grouped together based on their visual characteristics, rather than abstract relationships like functionality—e.g., we eliminate nodes that do not convey visual information such as “covering”. Second, nodes of similar specificity should be at the same distance from the

---

<sup>3</sup>In programming languages, this is known as “the diamond problem” or “the Deadly Diamond of Death” [Mar97].

root, irrespective of how detailed their categorization within WordNet is—for instance, we placed “dog” at the same level as “cat” and “flower”, even though the “dog” sub-tree in WordNet is much larger. Finally, we removed a number of ImageNet classes that did not naturally fit into the hierarchy. The resulting hierarchy, presented in Appendix A.4, contains nodes of comparable granularity at the same level. Moreover, as a result of this process, each node ends up having a single parent and thus the resulting hierarchy is a tree.

## 4.2 Creating BREEDS tasks

Once the modified version of the WordNet hierarchy is in place, BREEDS tasks can be created in an automated manner. Specifically, we first choose the desired granularity of the task by specifying the distance from the root (“entity”) and retrieving all superclasses at that distance in a top-down manner. Each resulting superclass corresponds to a subtree of our hierarchy, with ImageNet classes as its leaves. Note that these superclasses are roughly of the same specificity, due to our hierarchy restructuring process. Then, we randomly sample a fixed number of subclasses for each superclass to produce a balanced dataset (omitting superclasses with an insufficient number of subclasses). Finally, as described in Section 3, we randomly split these subclasses into the source and target domain.<sup>4</sup>

For our analysis, we create four tasks, presented in Table 1, based on different levels/parts of the hierarchy. To illustrate what the corresponding subpopulation shifts look like, we also present (random) image samples for a subset of the tasks in Figure 2. Note that while we focus on the tasks in Table 1 in our study, our methodology readily enables us to create other variants of these tasks in an automated manner.

| Name          | Subtree            | Level | Subpopulations | Examples              |
|---------------|--------------------|-------|----------------|-----------------------|
| ENTITY-13     | “entity” (root)    | 3     | 20             | “mammal”, “appliance” |
| ENTITY-30     | “entity” (root)    | 4     | 8              | “fruit”, “carnivore”  |
| LIVING-17     | “living thing”     | 5     | 4              | “ape”, “bear”         |
| NON-LIVING-26 | “non-living thing” | 5     | 4              | “fence”, “ball”       |

Table 1: BREEDS benchmarks constructed using ImageNet. “Level” indicates the depth of the superclasses in the class hierarchy (task granularity). The number of “subpopulations” (per superclass) is fixed across superclasses to ensure a balanced dataset. We can also construct specialized tasks, by focusing on subtrees in the hierarchy, e.g., only living (LIVING-17) or non-living (NON-LIVING-26) objects. Datasets are named based on the root of the subtree and the resulting number of superclasses they end up containing.

**BREEDS benchmarks beyond ImageNet.** It is worth noting that the methodology we described is not restricted to ImageNet and can be readily applied to other datasets as well. The only requirement is that we have access to a semantic grouping of the dataset classes, which is the case for many popular vision datasets—e.g., CIFAR-100 [Kri09], Pascal-VOC [Eve+10], OpenImages [Kuz+18], COCO-Stuff [CUF18]. Moreover, even when a class hierarchy is entirely absent, the needed semantic class grouping can be manually constructed with relatively little effort (proportional to the number of classes, not the number of datapoints).

## 4.3 Calibrating BREEDS benchmarks via human studies

For a distribution shift benchmark to be meaningful, it is essential that the source and target domains capture the same high-level task—otherwise generalizing from one domain to the other would be impossible. To ensure that this is the case for the BREEDS task, we assess how significant the resulting distribution shifts are for human annotators (crowd-sourced via MTurk).

**Annotator task.** To obtain meaningful performance estimates, it is crucial that annotators perform the task based only *on the visual content of the images*, without leveraging prior knowledge of the visual world. To

<sup>4</sup>We also consider more benign or adversarial subpopulation splits for these tasks in Appendix B.2.1.

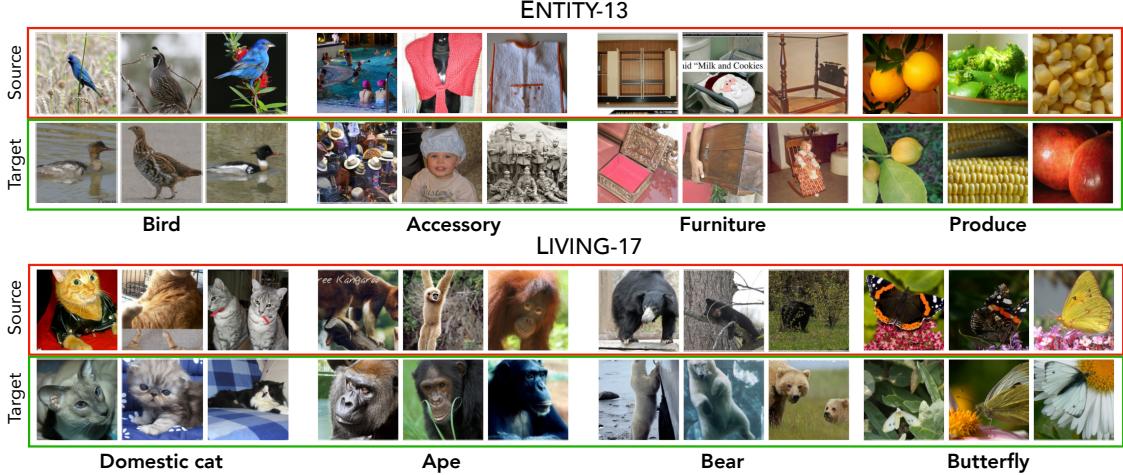


Figure 2: Sample images from random object categories for the ENTITY-13 and LIVING-17 tasks. For each task, the top and bottom row correspond to the source and target distributions respectively.

achieve this, we design the following annotation task. First, annotators are shown images from the source domain, grouped by superclass, without being aware of the superclass name (i.e., the object grouping it corresponds to). Then, they are presented with images from the target domain and are asked to assign each of them to one of the groups. For simplicity, we only present two random superclasses at a time, effectively simulating binary classification. Annotator accuracy can be measured directly as the fraction of images that they assign to the superclass to which these images belong. We perform this experiment for each of the BREEDS tasks constructed in Section 4.2. As a point of comparison, we repeat this experiment without subpopulation shift (test images are sampled from the source domain) and for the superclasses constructed by Huh, Agrawal, and Efros [HAE16] using the WordNet hierarchy directly (cf. Appendix A.5).

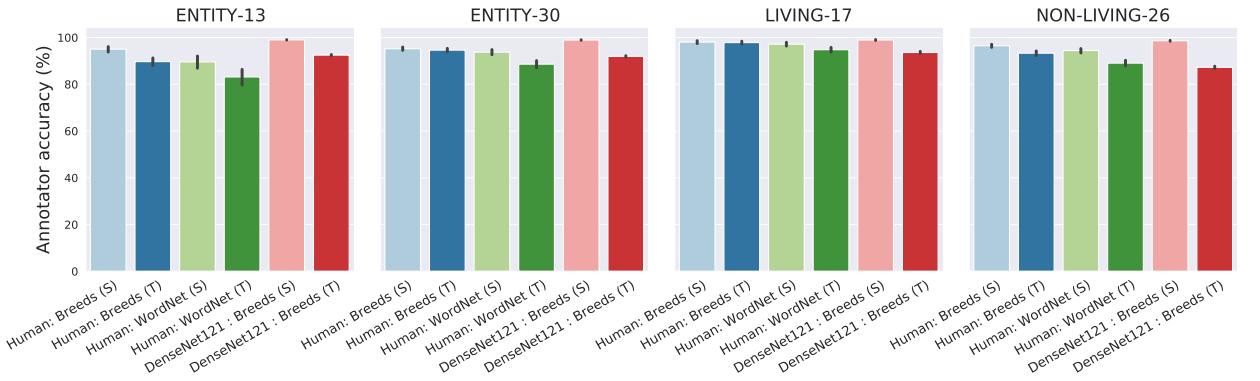


Figure 3: Human performance on (binary) BREEDS tasks. Annotators are provided with labeled images from the source distribution for a pair of (undisclosed) superclasses, and asked to classify samples from the target domain ('T') into one of the two groups. As a baseline we also measure annotator performance without subpopulation shift (i.e., on test images drawn from the source domain, 'S') and equivalent tasks created via the original WordNet hierarchy (cf. Appendix A.5). We can observe that across all tasks, annotators are fairly robust to subpopulation shift. Further, annotators consistently perform better on BREEDS task compared to those based on WordNet directly—indicating that our modified class hierarchy is indeed better calibrated for object recognition. (We discuss model performance in Section 5.)

**Human performance.** We find that, across all tasks, annotators perform well on unseen data from the source domain, as expected. More importantly, annotators also appear to be quite robust to subpopulation

shift, experiencing only a small accuracy drop between the source and target domains (cf. Figure 4). This is particularly prominent in the case of ENTITY-30 and LIVING-17 where the difference in source and target accuracy is within the confidence interval. This indicates that the source and target domains are indeed perceptually similar for humans, making these benchmarks suitable for studying model robustness. Finally, across all benchmarks, we observe that annotators perform better on BREEDS tasks, as compared to their WordNet equivalents—even on samples from the source domain. This indicates that our modified ImageNet class hierarchy is indeed better aligned with the underlying visual object recognition task.

## 5 Evaluating Model Performance under Subpopulation Shift

We can now use our suite of BREEDS tasks as a testbed for assessing model robustness to subpopulation shift as well as gauging the effectiveness of various train-time robustness interventions. Specifics of the evaluation setup and additional experimental results are provided in Appendices A.6 and B.2.

### 5.1 Standard training

We start by evaluating the performance of various model architectures trained in the standard fashion: empirical risk minimization (ERM) on the source distribution (cf. Appendix A.6.1). While these models perform well on unseen inputs from the domain they are trained on, i.e., they achieve high *source accuracy*, they suffer a considerable drop in accuracy under these subpopulation shifts—more than 30% in most cases (cf. Figure 4). At the same time, models that are more *accurate* on the source domain also appear to be more *robust* to distribution shift. Specifically, the fraction of source accuracy that is preserved in the target domain is typically increasing with source accuracy. (If this were not the case, i.e., the accuracy of all models dropped by a constant fraction under distribution shift, the target accuracy would match the baseline in Figure 4.) This indicates that, while models are quite brittle to subpopulation shift, improvements in source accuracy *do* correlate with models generalizing better to variations in testing conditions. Note that model accuracies are not directly comparable across benchmarks, due to the presence of multiple conflating factors. On one hand, more fine-grained tasks present a smaller subpopulation shift (subclasses are semantically closer). On the other hand, the number of classes and training inputs per class changes significantly, making the task harder.

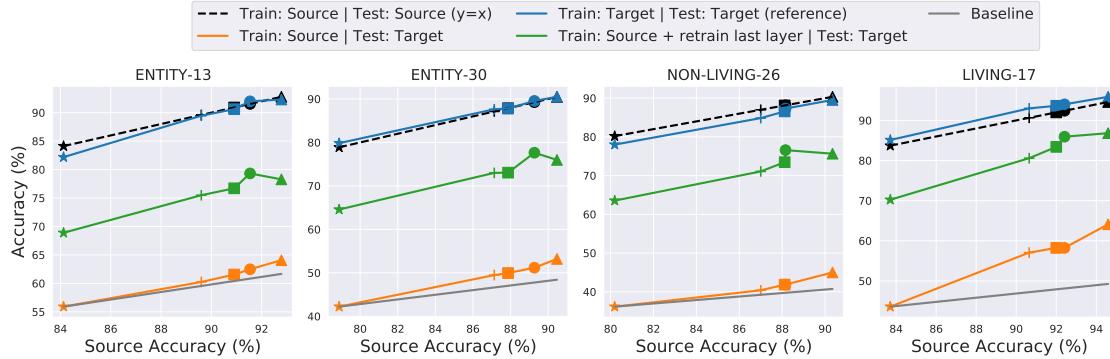


Figure 4: Robustness of standard models to BREEDS subpopulation shifts. For each of the four tasks, we plot the accuracy of different (source domain-trained) model architectures (denoted by different symbols) on the target domain as a function of the source accuracy (which is typically high). We find that model accuracy drops significantly between domains (*orange* vs. *dashed line*). Still, models that are more accurate on the source domain seem to also be more robust (the improvements exceed the baseline (*grey*) which would correspond to a constant accuracy drop across models, i.e.,  $\frac{\text{source acc}}{\text{target acc}} = \text{constant}$  based on AlexNet). Moreover, the drop in model performance on the target domain can be reduced by retraining the final model layer with data from that domain (*green*). However, a non-trivial drop persists compared to both the original source accuracy, and target accuracy of models trained directly (end-to-end) on the target domain (*blue*).

**Models vs. Humans.** We compare the best performing model (DenseNet-121 in this case) to our previously obtained human baselines in Figure 3. To allow for a fair comparison, model accuracy is measured on pairwise superclass classification tasks (cf. Appendix A.6). We observe that models do exceedingly well on unseen samples from the source domain—significantly outperforming annotators under our task setup. At the same time, models also appear to be more brittle, performing worse than humans on the target domain of these binary BREEDS tasks, despite their higher source accuracy.

**Adapting models to the target domain.** Finally, we focus on the intermediate data representations learned by these models, aiming to assess how suitable they are for distinguishing classes in the target domain. To assess this, we retrain the last (fully-connected) layer of models trained on the source domain with data from the target domain. We find that the target accuracy of these models increases significantly after retraining, indicating that the learned representations indeed generalize to the target domain. However, we cannot match the accuracy of models trained directly (end-to-end) on the target domain—see Figure 4—demonstrating that there is significant room for improvement.

## 5.2 Robustness interventions

We now turn our attention to existing methods for decreasing model sensitivity to specific synthetic perturbations. Our goal is to assess if these methods enhance model robustness to subpopulation shift too. Concretely, we consider the following families of interventions (cf. Appendix A.6.3 for details):

- **Adversarial training:** Enhances robustness to worst-case  $\ell_p$ -bounded perturbations (in our case  $\ell_2$ ) by training models against a projected gradient descent (PGD) adversary [Mad+18].
- **Stylized Training:** Encourages models to rely more on shape rather than texture by training them on a stylized version of ImageNet [Gei+19].
- **Random noise:** Improves model robustness to data corruptions by incorporating them as data augmentations during training—we focus on Gaussian noise and Erase noise [Zho+20], i.e., randomly obfuscating a block of the image.

Note that these methods can be viewed as ways of imposing a prior on the features that the model relies on [HM17; Gei+19; Eng+19b]. That is, by rendering certain features ineffective during training (e.g., texture) they incentivize the model to utilize alternative features for its predictions (e.g., shape). Since different families of features may correlate differently with class labels in the target domain, the aforementioned interventions could significantly impact model robustness to subpopulation shift.

**Relative accuracy.** To measure the impact of these interventions, we will focus on the models’ *relative accuracy*—the ratio of target accuracy to source accuracy. This metric accounts for the fact that train-time interventions can impact model accuracy on the source domain itself. By measuring relative performance, we are able to compare different training methods on an equal footing.

We find that robustness interventions *do* have a small, yet non-trivial, impact on the robustness of a particular model architecture to subpopulation shift—see Figure 5. Specifically, for the case of adversarial training and erase noise, models often retain a larger fraction of their accuracy to the target domain compared to standard training, hence lying on the Pareto frontier of a robustness-accuracy trade-off. In fact, for some of the models trained with these interventions, the target accuracy is slightly higher than models obtained via standard training, even without adjusting for their lower source accuracy (raw accuracies for all methods are in Appendix B.2.2). Nonetheless, it is important to note that none of these method offer significant subpopulation robustness—relative accuracy is not improved by more than a few percentage points.

**Adapting models to the target domain.** The impact of these interventions is more pronounced if we consider the target accuracy of these models after their last layer has been retrained on data from the target domain—see Figure 6. In particular, we observe that for adversarially robust models, retraining significantly boosts accuracy on the target domain—e.g., in the case of LIVING-17 it is almost comparable to the initial accuracy on the source domain. This indicates that the feature priors imposed by these interventions

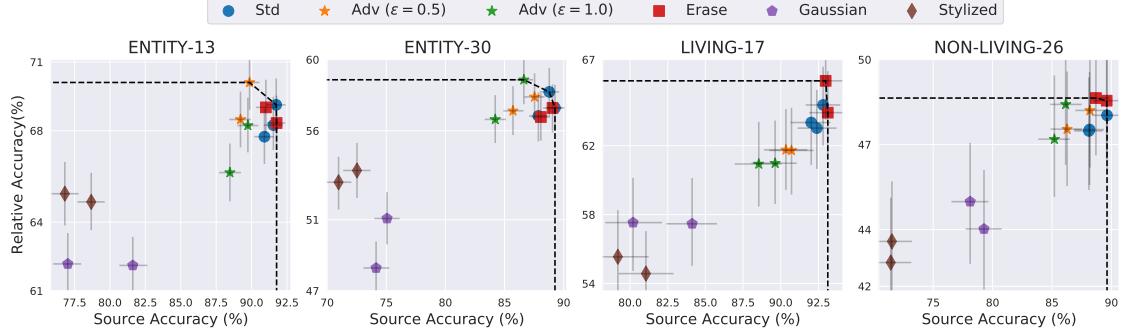


Figure 5: Effect of train-time interventions on model robustness to subpopulation shift. We measure model performance in terms of *relative accuracy*—i.e., the ratio between its target and source accuracies. This allows us to visualize the accuracy-robustness trade-off along with the corresponding Pareto frontier (*dashed*). (Also shown are 95% confidence intervals computed via bootstrapping.) We observe that some of these interventions do improve model robustness to subpopulation shift by a small amount—specifically, erase noise and adversarial training—albeit sometimes at the cost of source accuracy.

incentivize models to learn representations that generalize better to similar domains—in line with recent results of Utrera et al. [Utr+20] and Salman et al. [Sal+20]. Moreover, we observe that models trained on the stylized version of these datasets perform consistently worse, suggesting that texture might be an important feature for these tasks, especially in the presence of subpopulation shift. Finally, note that we did not perform an exhaustive exploration of the hyper-parameters used for these interventions (e.g.,  $\ell_2$ -norm)—it is possible that these results can be improved by additional tuning. For instance, we would expect that we can tune the magnitude of the Gaussian noise to achieve performance that is comparable to that of  $\ell_2$ -bounded adversarial training [For+19].

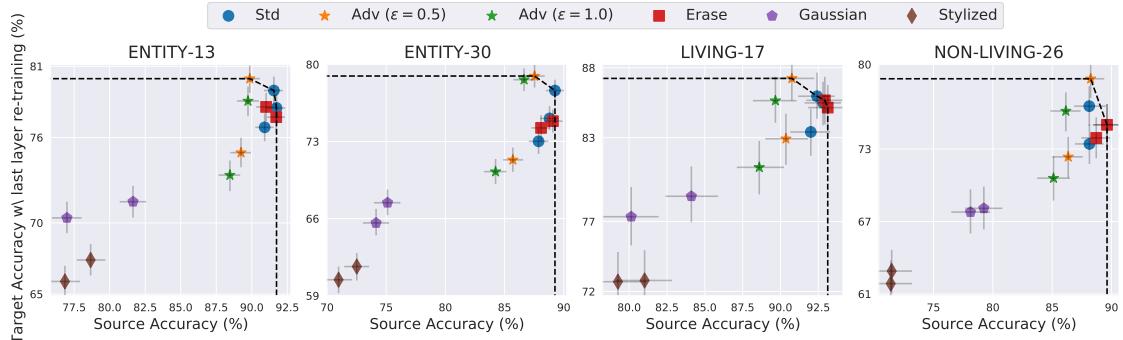


Figure 6: Target accuracy of models after they have been retrained (only the final linear layer) on data from the target domain (with 95% bootstrap confidence intervals). Models trained with robustness interventions often have higher target accuracy than standard models post retraining.

## 6 Additional Related Work

In Section 2, we discuss prior work that has directly focused on evaluating model robustness to distribution shift. We now provide an overview of other related work and its connections to our methodology.

**Distributional robustness.** Distribution shifts that are small with respect to some  $f$ -divergence have been studied in prior theoretical work [Ben+13; DGN16; EK18; ND16]. However, this notion of robustness is typically too pessimistic to capture realistic data variations [Hu+18]. Distributional robustness has also been

connected to causality [Mei18]: here, the typical approach is to inject spurious correlations into the dataset, and assess to what extent models rely on them for their predictions [HM17; Arj+19; Sag+20].

**Domain adaptation and transfer learning.** The goal here is to adapt models to the target domain with relatively few samples from it [Ben+07; Sae+10; GL15; Cou+16; Gon+16; Don+14; Sha+14]. In domain adaptation, the task is the same in both domains, while in transfer learning, the task itself could vary. In a similar vein, the field of *domain generalization* aims to generalize to samples from a different domain (e.g., from ClipArt to photos) by training on a number of explicitly annotated domains [MBS13; Li+17; Pen+19].

**Zero-shot learning.** Work in this domain focuses on learning to recognize previously unseen classes [LNH09; XSA17], typically described via a semantic embedding [LNH09; Mik+13; Soc+13; Fro+13; RT15]. This differs from our setup, where the focus is on generalization to unseen subpopulations for the *same* set of classes.

## 7 Conclusion

In this work, we develop a methodology for constructing large-scale subpopulation shift benchmarks. The motivation behind our BREEDS benchmarks is to test if models can generalize beyond the limited diversity of their training datasets—specifically, to novel data subpopulations. A major advantage of our approach is its generality. It can be applied to any dataset with a meaningful class structure—including tasks beyond classification (e.g., object detection) and domains other than computer vision (e.g., natural language processing). Moreover, the subpopulation shifts are induced in a manner that is both controlled and natural, without altering inputs synthetically or needing to collect new data.

We apply this approach to the ImageNet dataset to construct benchmarks of varying difficulty. We then demonstrate how these benchmarks can be used to assess model robustness and the efficacy of various train-time interventions. Further, we obtain human baselines for these tasks to both put model performance in context and validate that the corresponding subpopulation shifts do not significantly affect humans.

Overall, our results indicate that existing models still have a long way to go before they can fully tackle the BREEDS subpopulation shifts, even with robustness interventions. We thus believe that our methodology provides a useful framework for studying model robustness to distribution shift—an increasingly pertinent topic for real-world deployments of machine learning models.

## Acknowledgements

We thank Andrew Ilyas and Sam Park for helpful discussions.

Work supported in part by the NSF grants CCF-1553428, CNS-1815221, the Google PhD Fellowship, and the Microsoft Corporation. This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001120C0015.

Research was sponsored by the United States Air Force Research Laboratory and was accomplished under Cooperative Agreement Number FA8750-19-2-1000. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

## References

- [Arj+19] Martin Arjovsky et al. “Invariant risk minimization”. In: *arXiv preprint arXiv:1907.02893* (2019).
- [Ben+07] Shai Ben-David et al. “Analysis of representations for domain adaptation”. In: *Neural Information Processing Systems (NeurIPS)*. 2007.
- [Ben+13] Aharon Ben-Tal et al. “Robust solutions of optimization problems affected by uncertain probabilities”. In: *Management Science*. 2013.

- [BG18] Joy Buolamwini and Timnit Gebru. “Gender shades: Intersectional accuracy disparities in commercial gender classification”. In: *Conference on fairness, accountability and transparency (FAccT)*. 2018.
- [BVP18] Sara Beery, Grant Van Horn, and Pietro Perona. “Recognition in terra incognita”. In: *European Conference on Computer Vision (ECCV)*. 2018.
- [Cal+18] Sebastian Caldas et al. “Leaf: A benchmark for federated settings”. In: *arXiv preprint arXiv:1812.01097* (2018).
- [Cou+16] Nicolas Courty et al. “Optimal transport for domain adaptation”. In: *transactions on pattern analysis and machine intelligence*. 2016.
- [CUF18] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. “COCO-Stuff: Thing and stuff classes in context”. In: *Computer vision and pattern recognition (CVPR)*, 2018. 2018.
- [Den+09] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *computer vision and pattern recognition (CVPR)*. 2009.
- [DGN16] John Duchi, Peter Glynn, and Hongseok Namkoong. “Statistics of robust optimization: A generalized empirical likelihood approach”. In: *Mathematics of Operations Research* (2016).
- [DN18] John Duchi and Hongseok Namkoong. “Learning models with uniform performance via distributionally robust optimization”. In: *arXiv preprint arXiv:1810.08750*. 2018.
- [Don+14] Jeff Donahue et al. “Decaf: A deep convolutional activation feature for generic visual recognition”. In: *International conference on machine learning (ICML)*. 2014.
- [Dwo+12] Cynthia Dwork et al. “Fairness through awareness”. In: *innovations in theoretical computer science conference (ITCS)*. 2012.
- [EK18] Peyman Mohajerin Esfahani and Daniel Kuhn. “Data-driven distributionally robust optimization using the Wasserstein metric: Performance guarantees and tractable reformulations”. In: *Mathematical Programming* (2018).
- [Eng+19a] Logan Engstrom et al. “Exploring the Landscape of Spatial Robustness”. In: *International Conference on Machine Learning (ICML)*. 2019.
- [Eng+19b] Logan Engstrom et al. “Learning Perceptually-Aligned Representations via Adversarial Robustness”. In: *ArXiv preprint arXiv:1906.00945*. 2019.
- [Eng+20] Logan Engstrom et al. “Identifying Statistical Bias in Dataset Replication”. In: *International Conference on Machine Learning (ICML)*. 2020.
- [Eve+10] M. Everingham et al. “The Pascal Visual Object Classes (VOC) Challenge”. In: *International Journal of Computer Vision*. 2010.
- [FF15] Alhussein Fawzi and Pascal Frossard. “Manitest: Are classifiers really invariant?” In: *British Machine Vision Conference (BMVC)*. 2015.
- [FMF16] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. “Robustness of classifiers: from adversarial to random noise”. In: *Advances in Neural Information Processing Systems*. 2016.
- [For+19] Nic Ford et al. “Adversarial Examples Are a Natural Consequence of Test Error in Noise”. In: *arXiv preprint arXiv:1901.10513*. 2019.
- [Fro+13] Andrea Frome et al. “Devise: A deep visual-semantic embedding model”. In: *neural information processing systems (NeurIPS)*. 2013.
- [Gei+19] Robert Geirhos et al. “ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness.” In: *International Conference on Learning Representations*. 2019.
- [GL15] Yaroslav Ganin and Victor Lempitsky. “Unsupervised domain adaptation by backpropagation”. In: 2015.
- [Gon+16] Mingming Gong et al. “Domain adaptation with conditional transferable components”. In: *International conference on machine learning (ICML)*. 2016.

- [HAE16] Minyoung Huh, Pulkit Agrawal, and Alexei A Efros. "What makes ImageNet good for transfer learning?" In: *arXiv preprint arXiv:1608.08614* (2016).
- [Has+18] Tatsunori Hashimoto et al. "Fairness Without Demographics in Repeated Loss Minimization". In: *International Conference on Machine Learning (ICML)*. 2018.
- [HD19] Dan Hendrycks and Thomas G. Dietterich. "Benchmarking Neural Network Robustness to Common Corruptions and Surface Variations". In: *International Conference on Learning Representations (ICLR)*. 2019.
- [HM17] Christina Heinze-Deml and Nicolai Meinshausen. "Conditional variance penalties and domain shift robustness". In: *arXiv preprint arXiv:1710.11469* (2017).
- [Hu+18] Weihua Hu et al. "Does distributionally robust supervised learning give robust classifiers?" In: *International Conference on Machine Learning (ICML)*. 2018.
- [JTJ17] David Jurgens, Yulia Tsvetkov, and Dan Jurafsky. "Incorporating dialectal variability for socially equitable language identification". In: *Association for Computational Linguistics (ACL)*. 2017.
- [Kan+19] Daniel Kang et al. "Testing Robustness Against Unforeseen Adversaries". In: *ArXiv preprint arxiv:1908.08016*. 2019.
- [KHA99] Mark G Kelly, David J Hand, and Niall M Adams. "The impact of changing populations on classifier performance". In: *international conference on Knowledge discovery and data mining (SIGKDD)*. 1999.
- [Kho+12] Aditya Khosla et al. "Undoing the damage of dataset bias". In: *European Conference on Computer Vision (ECCV)*. 2012.
- [KMR17] Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. "Inherent Trade-Offs in the Fair Determination of Risk Scores". In: *Innovations in Theoretical Computer Science (ITCS)*. 2017.
- [Kri09] Alex Krizhevsky. "Learning Multiple Layers of Features from Tiny Images". In: *Technical report*. 2009.
- [Kuz+18] Alina Kuznetsova et al. "The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale". In: *arXiv preprint arXiv:1811.00982* (2018).
- [Li+17] Da Li et al. "Deeper, broader and artier domain generalization". In: *international conference on computer vision (ICCV)*. 2017.
- [LNH09] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. "Learning to detect unseen object classes by between-class attribute transfer". In: *Computer Vision and Pattern Recognition (CVPR)*. 2009.
- [M+15] Nicolai Meinshausen, Peter Bühlmann, et al. "Maximin effects in inhomogeneous large-scale data". In: *The Annals of Statistics* (2015).
- [Mad+18] Aleksander Madry et al. "Towards deep learning models resistant to adversarial attacks". In: *International Conference on Learning Representations (ICLR)*. 2018.
- [Mar97] Robert C Martin. "Java and C++: A critical comparison". In: (1997).
- [MBS13] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. "Domain generalization via invariant feature representation". In: *International Conference on Machine Learning (ICML)*. 2013.
- [Mei18] Nicolai Meinshausen. "Causality from a distributional robustness point of view". In: *Data Science Workshop (DSW)*. 2018.
- [Mik+13] Tomas Mikolov et al. "Distributed representations of words and phrases and their compositionality". In: *Advances in neural information processing systems*. 2013, pp. 3111–3119.
- [Mil95] George A Miller. "WordNet: a lexical database for English". In: *Communications of the ACM* (1995).
- [Mor+12] Jose G Moreno-Torres et al. "A unifying view on dataset shift in classification". In: *Pattern recognition* (2012).

- [ND16] Hongseok Namkoong and John C Duchi. "Stochastic gradient methods for distributionally robust optimization with f-divergences". In: *neural information processing systems (NeurIPS)*. 2016.
- [Ore+19] Yonatan Oren et al. "Distributionally Robust Language Modeling". In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2019.
- [Pen+19] Xingchao Peng et al. "Moment matching for multi-source domain adaptation". In: *International Conference on Computer Vision (ICCV)*. 2019.
- [Pon+06] Jean Ponce et al. "Dataset issues in object recognition". In: *Toward category-level object recognition*. 2006.
- [Qui+09] Joaquin Quionero-Candela et al. *Dataset shift in machine learning*. The MIT Press, 2009.
- [Rec+19] Benjamin Recht et al. "Do ImageNet Classifiers Generalize to ImageNet?" In: *International Conference on Machine Learning (ICML)*. 2019.
- [RT15] Bernardino Romera-Paredes and Philip Torr. "An embarrassingly simple approach to zero-shot learning". In: *International Conference on Machine Learning (ICML)*. 2015.
- [Rus+15] Olga Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)*. 2015.
- [Sae+10] Kate Saenko et al. "Adapting visual category models to new domains". In: *European conference on computer vision (ECCV)*. 2010.
- [Sag+20] Shiori Sagawa et al. "Distributionally Robust Neural Networks for Group Shifts: On the Importance of Regularization for Worst-Case Generalization". In: *International Conference on Learning Representations*. 2020.
- [Sal+20] Hadi Salman et al. "Do Adversarially Robust ImageNet Models Transfer Better?" In: *arXiv preprint arXiv:2007.08489*. 2020.
- [SG86] Jeffrey C Schlimmer and Richard H Granger. "Beyond Incremental Processing: Tracking Concept Drift." In: *AAAI*. 1986.
- [Sha+14] Ali Sharif Razavian et al. "CNN features off-the-shelf: an astounding baseline for recognition". In: *conference on computer vision and pattern recognition (CVPR) workshops*. 2014.
- [Sha+19] Vaishaal Shankar et al. "Do Image Classifiers Generalize Across Time?" In: *arXiv preprint arXiv:1906.02168* (2019).
- [Shi00] Hideyoshi Shimodaira. "Improving predictive inference under covariate shift by weighting the log-likelihood function". In: *Journal of statistical planning and inference* (2000).
- [SK12] Masashi Sugiyama and Motoaki Kawanabe. *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*. MIT press, 2012.
- [SKM07] Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert MÃžller. "Covariate shift adaptation by importance weighted cross validation". In: *Journal of Machine Learning Research (JMLR)* (2007).
- [Soc+13] Richard Socher et al. "Zero-shot learning through cross-modal transfer". In: *neural information processing systems (NeurIPS)*. 2013.
- [Sze+14] Christian Szegedy et al. "Intriguing properties of neural networks". In: *International Conference on Learning Representations (ICLR)*. 2014.
- [Tao+20] Rohan Taori et al. "Measuring Robustness to Natural Distribution Shifts in Image Classification". In: *arXiv preprint arXiv:2007.00644* (2020).
- [TE11] Antonio Torralba and Alexei A Efros. "Unbiased look at dataset bias". In: *CVPR 2011*. 2011.
- [Tsi+20] Dimitris Tsipras et al. "From ImageNet to Image Classification: Contextualizing Progress on Benchmarks". In: *International Conference on Machine Learning (ICML)*. 2020.
- [TT14] Tatiana Tommasi and Tinne Tuytelaars. "A testbed for cross-dataset analysis". In: *European Conference on Computer Vision (ECCV)*. 2014.
- [Utr+20] Francisco Utrera et al. "Adversarially-Trained Deep Nets Transfer Better". In: *ArXiv preprint arXiv:2007.05869*. 2020.

- [WK93] Gerhard Widmer and Miroslav Kubat. "Effective learning in dynamic environments by explicit context tracking". In: *European Conference on Machine Learning*. 1993.
- [XSA17] Yongqin Xian, Bernt Schiele, and Zeynep Akata. "Zero-shot learning-the good, the bad and the ugly". In: *Computer Vision and Pattern Recognition (CVPR)*. 2017.
- [Zho+20] Zhun Zhong et al. "Random Erasing Data Augmentation." In: *AAAI*. 2020.

## A Experimental Setup

### A.1 Dataset

We perform our analysis on the ILSVRC2012 dataset [Rus+15]. This dataset contains a thousand classes from the ImageNet dataset [Den+09] with an independently collected validation set. The classes are part of the broader hierarchy, WordNet [Mil95], through which words are organized based on their semantic meaning. We use this hierarchy as a starting point of our investigation but modify it as described in Appendix A.4.

For all the BREEDS superclass classification tasks, the train and validation sets are obtained by aggregating the train and validation sets of the descendant ImageNet classes (i.e., subpopulations). Specifically, for a given subpopulation, the training and test splits from the original ImageNet dataset are used as is.

### A.2 WordNet issues

As discussed in Section 4, WordNet is a semantic rather than a visual hierarchy. That is, object classes are arranged based on their meaning rather than their visual appearance. Thus, using intermediate nodes for a visual object recognition task is not straightforward. To illustrate this, we examine a sample superclass grouping created by Huh, Agrawal, and Efros [HAE16] via automated bottom-up clustering in Table 2.

| Superclass                | Random ImageNet classes   |
|---------------------------|---|
| <b>instrumentality</b>    | fire engine, basketball, electric fan, wok, thresher, horse cart, harvester, balloon, racket, can opener, carton, gong, unicycle, toilet seat, carousel, hard disc, cello, mousetrap, neck brace, barrel  |
| <b>man-made structure</b> | beacon, yurt, picket fence, barbershop, fountain, steel arch bridge, library, cinema, stone wall, worm fence, palace, suspension bridge, planetarium, monastery, mountain tent, sliding door, dam, bakery, megalith, pedestal   |
| <b>covering</b>           | window shade, vestment, running shoe, diaper, sweatshirt, breastplate, shower curtain, shoji, miniskirt, knee pad, apron, pajama, military uniform, theater curtain, jersey, football helmet, book jacket, bow tie, suit, cloak   |
| <b>commodity</b>          | espresso maker, maillot, iron, bath towel, lab coat, bow tie, washer, jersey, mask, waffle iron, mortarboard, diaper, bolo tie, seat belt, cowboy hat, wig, knee pad, vacuum, microwave, abaya  |
| <b>organism</b>           | thunder snake, stingray, grasshopper, barracouta, Newfoundland, Mexican hairless, Welsh springer spaniel, bluetick, golden retriever, keeshond, African chameleon, jacamar, water snake, Staffordshire bullterrier, Old English sheepdog, pelican, sea lion, wire-haired fox terrier, flamingo, green mamba |
| <b>produce</b>            | spaghetti squash, fig, cardoon, mashed potato, pineapple, zucchini, broccoli, cauliflower, butternut squash, custard apple, pomegranate, strawberry, Granny Smith, lemon, head cabbage, artichoke, cucumber, banana, bell pepper, acorn squash  |

Table 2: Superclasses constructed by Huh, Agrawal, and Efros [HAE16] via bottom-up clustering of WordNet to obtain 36 superclasses—for brevity, we only show superclasses with at least 20 ImageNet classes each.

First, we can notice that these superclasses have vastly different granularities. For instance, “organism” contains the entire animal kingdom, hence being much broader than “produce”. Moreover, “covering” is rather abstract class, and hence its subclasses often share little visual similarity (e.g., “window shade”, “pajama”). Finally, due to the abstract nature of these superclasses, a large number of subclasses overlap—“covering” and “commodity” share 49 ImageNet descendants.

### A.3 Manual calibration

In order to allow for efficient and automated creation of superclasses that are suitable for visual recognition, we modified the WordNet hierarchy by applying the following operations:

- *Collapse node*: Delete a node from the hierarchy and add edges from each parent to each child. Allows us to remove redundant or overly specific categorization while preserving the overall structure.
- *Insert node above*: Add a dummy parent to push a node further down the hierarchy. Allows us to ensure that nodes of similar granularity are at the same level.
- *Delete node*: Remove a node and all of its edges. Used to remove abstract nodes that do not reveal visual characteristics.
- *Add edge*: Connect a node to a parent. Used to reassign the children of nodes deleted by the operation above.

We manually examined the hierarchy and implemented these actions in order to produce superclasses that are calibrated for classification. The principles we followed are outlined in Section 4 while the full hierarchy can be explored using the notebooks provided with the hierarchy.<sup>5</sup>

### A.4 Resulting hierarchy

The parameters for constructing the BREEDS benchmarks (hierarchy level, number of subclasses, and tree root) are given in Table 1. The resulting tasks—obtained by sampling disjoint ImageNet classes (i.e., subpopulations) for the source and target domain—are shown in Tables 3, 4, 5, and 6. Recall that for each superclass we randomly sample a fixed number of subclasses per superclass to ensure that the dataset is approximately balanced.

---

<sup>5</sup><https://github.com/MadryLab/BREEDS-Benchmarks>

| Superclass                | Source   | Target   |
|---------------------------|--|--|
| <b>garment</b>            | trench coat, abaya, gown, poncho, military uniform, jersey, cloak, bikini, miniskirt, swimming trunks  | lab coat, brassiere, hoopskirt, cardigan, pajama, academic gown, apron, diaper, sweatshirt, sarong   |
| <b>bird</b>               | African grey, bee eater, coucal, American coot, indigo bunting, king penguin, spoonbill, limpkin, quail, kite                                | prairie chicken, red-breasted merganser, albatross, water ouzel, goose, oystercatcher, American egret, hen, lorikeet, ruffed grouse                |
| <b>reptile</b>            | Gila monster, agama, triceratops, African chameleon, thunder snake, Indian cobra, green snake, mud turtle, water snake, loggerhead           | sidewinder, leatherback turtle, boa constrictor, garter snake, terrapin, box turtle, ringneck snake, rock python, American chameleon, green lizard |
| <b>arthropod</b>          | rock crab, black and gold garden spider, tiger beetle, black widow, barn spider, leafhopper, ground beetle, fiddler crab, bee, walking stick | cabbage butterfly, admiral, lacewing, trilobite, sulphur butterfly, cicada, garden spider, leaf beetle, long-horned beetle, fly                    |
| <b>mammal</b>             | Siamese cat, ibex, tiger, hippopotamus, Norwegian elkhound, dugong, colobus, Samoyed, Persian cat, Irish wolfhound                           | English setter, llama, lesser panda, armadillo, indri, giant schnauzer, pug, Doberman, American Staffordshire terrier, beagle                      |
| <b>accessory</b>          | bib, feather boa, stole, plastic bag, bathing cap, cowboy boot, necklace, crash helmet, gasmask, maillot                                     | hair slide, umbrella, pickelhaube, mitten, sombrero, shower cap, sock, running shoe, mortarboard, handkerchief                                     |
| <b>craft</b>              | catamaran, speedboat, fireboat, yawl, airliner, container ship, liner, trimaran, space shuttle, aircraft carrier                             | schooner, gondola, canoe, wreck, warplane, balloon, submarine, pirate, lifeboat, airship   |
| <b>equipment</b>          | volleyball, notebook, basketball, handheld computer, tripod, projector, barbell, monitor, croquet ball, balance beam                         | cassette player, snorkel, horizontal bar, soccer ball, racket, baseball, joystick, microphone, tape player, reflex camera                          |
| <b>furniture</b>          | wardrobe, toilet seat, file, mosquito net, four-poster, bassinet, chiffonier, folding chair, fire screen, shoji                              | studio couch, throne, crib, rocking chair, dining table, park bench, chest, window screen, medicine chest, barber chair                            |
| <b>instrument</b>         | upright, padlock, lighter, steel drum, parking meter, cleaver, syringe, abacus, scale, corkscrew   | maraca, saltshaker, magnetic compass, accordion, digital clock, screw, can opener, odometer, organ, screwdriver                                    |
| <b>man-made structure</b> | castle, bell cote, fountain, planetarium, traffic light, breakwater, cliff dwelling, monastery, prison, water tower                          | suspension bridge, worm fence, turnstile, tile roof, beacon, street sign, maze, chain-link fence, bakery, drilling platform                        |
| <b>wheeled vehicle</b>    | snowplow, trailer truck, racer, shopping cart, unicycle, motor scooter, passenger car, minibus, jeep, recreational vehicle                   | jinrikisha, golfcart, tow truck, ambulance, bullet train, fire engine, horse cart, streetcar, tank, Model T  |
| <b>produce</b>            | broccoli, corn, orange, cucumber, spaghetti squash, butternut squash, acorn squash, cauliflower, bell pepper, fig                            | pomegranate, mushroom, strawberry, lemon, head cabbage, Granny Smith, hip, ear, banana, artichoke  |

Table 3: Superclasses used for the ENTITY-13 task, along with the corresponding subpopulations that comprise the source and target domains.

| Superclass                  | Source  | Target   |
|-----------------------------|---|--|
| <b>serpentes</b>            | green mamba, king snake, garter snake, thunder snake                | boa constrictor, green snake, ringneck snake, rock python  |
| <b>passerine</b>            | goldfinch, brambling, water ouzel, chickadee                        | magpie, house finch, indigo bunting, bulbul                |
| <b>saurian</b>              | alligator lizard, Gila monster, American chameleon, green lizard    | Komodo dragon, African chameleon, agama, banded gecko      |
| <b>arachnid</b>             | harvestman, barn spider, scorpion, black widow                      | wolf spider, black and gold garden spider, tick, tarantula |
| <b>aquatic bird</b>         | albatross, red-backed sandpiper, crane, white stork                 | goose, dowitcher, limpkin, drake                           |
| <b>crustacean</b>           | crayfish, spiny lobster, hermit crab, Dungeness crab                | king crab, rock crab, American lobster, fiddler crab       |
| <b>carnivore</b>            | Italian greyhound, black-footed ferret, Bedlington terrier, basenji | flat-coated retriever, otterhound, Shih-Tzu, Boston bull   |
| <b>insect</b>               | lacewing, fly, grasshopper, sulphur butterfly                       | long-horned beetle, leafhopper, dung beetle, admiral       |
| <b>ungulate</b>             | llama, gazelle, zebra, ox   | hog, hippopotamus, hartebeest, warthog                     |
| <b>primate</b>              | baboon, howler monkey, Madagascar cat, chimpanzee                   | siamang, indri, capuchin, patas                            |
| <b>bony fish</b>            | coho, tench, lionfish, rock beauty                                  | sturgeon, puffer, eel, gar                                 |
| <b>barrier</b>              | breakwater, picket fence, turnstile, bannister                      | chainlink fence, stone wall, dam, worm fence               |
| <b>building</b>             | bookshop, castle, mosque, butcher shop                              | grocery store, toyshop, palace, beacon                     |
| <b>electronic equipment</b> | printer, pay-phone, microphone, computer keyboard                   | modem, cassette player, monitor, dial telephone            |
| <b>footwear</b>             | clog, Loafer, maillot, running shoe                                 | sandal, knee pad, cowboy boot, Christmas stocking          |
| <b>garment</b>              | academic gown, apron, miniskirt, fur coat                           | jean, vestment, sarong, swimming trunks                    |
| <b>headdress</b>            | pickelhaube, hair slide, shower cap, bonnet                         | bathing cap, cowboy hat, bearskin, crash helmet            |
| <b>home appliance</b>       | washer, microwave, Crock Pot, vacuum                                | toaster, espresso maker, space heater, dishwasher          |
| <b>kitchen utensil</b>      | measuring cup, cleaver, coffeepot, spatula                          | frying pan, cocktail shaker, tray, cauldron                |
| <b>measuring instrument</b> | digital watch, analog clock, parking meter, magnetic compass        | barometer, wall clock, hourglass, digital clock            |
| <b>motor vehicle</b>        | limousine, school bus, moped, convertible                           | trailer truck, beach wagon, police van, garbage truck      |
| <b>musical instrument</b>   | French horn, maraca, grand piano, upright                           | acoustic guitar, organ, electric guitar, violin            |
| <b>neckwear</b>             | feather boa, neck brace, bib, Windsor tie                           | necklace, stole, bow tie, bolo tie                         |

|                         |  |   |
|-------------------------|--|---|
| <b>sports equipment</b> | ski, dumbbell, croquet ball, racket                  | rugby ball, balance beam, horizontal bar, tennis ball |
| <b>tableware</b>        | mixing bowl, water jug, beer glass, water bottle     | goblet, wine bottle, coffee mug, plate                |
| <b>tool</b>             | quill, combination lock, padlock, screw              | fountain pen, screwdriver, shovel, torch              |
| <b>vessel</b>           | container ship, lifeboat, aircraft carrier, trimaran | liner, wreck, catamaran, yawl                         |
| <b>dish</b>             | potpie, mashed potato, pizza, cheeseburger           | burrito, hot pot, meat loaf, hotdog                   |
| <b>vegetable</b>        | zucchini, cucumber, butternut squash, artichoke      | cauliflower, spaghetti squash, acorn squash, cardoon  |
| <b>fruit</b>            | strawberry, pineapple, jackfruit, Granny Smith       | buckeye, corn, ear, acorn                             |

Table 4: Superclasses used for the ENTITY-30 task, along with the corresponding subpopulations that comprise the source and target domains.

| Superclass          | Source                                  | Target                                 |
|---------------------|---|--|
| <b>salamander</b>   | eft, axolotl                            | common newt, spotted salamander        |
| <b>turtle</b>       | box turtle, leatherback turtle          | loggerhead, mud turtle                 |
| <b>lizard</b>       | whiptail, alligator lizard              | African chameleon, banded gecko        |
| <b>snake</b>        | night snake, garter snake               | sea snake, boa constrictor             |
| <b>spider</b>       | tarantula, black and gold garden spider | garden spider, wolf spider             |
| <b>grouse</b>       | ptarmigan, prairie chicken              | ruffed grouse, black grouse            |
| <b>parrot</b>       | macaw, lorikeet                         | African grey, sulphur-crested cockatoo |
| <b>crab</b>         | Dungeness crab, fiddler crab            | rock crab, king crab                   |
| <b>dog</b>          | bloodhound, Pekinese                    | Great Pyrenees, papillon               |
| <b>wolf</b>         | coyote, red wolf                        | white wolf, timber wolf                |
| <b>fox</b>          | grey fox, Arctic fox                    | red fox, kit fox                       |
| <b>domestic cat</b> | tiger cat, Egyptian cat                 | Persian cat, Siamese cat               |
| <b>bear</b>         | sloth bear, American black bear         | ice bear, brown bear                   |
| <b>beetle</b>       | dung beetle, rhinoceros beetle          | ground beetle, long-horned beetle      |
| <b>butterfly</b>    | sulphur butterfly, admiral              | cabbage butterfly, ringlet             |
| <b>ape</b>          | gibbon, orangutan                       | gorilla, chimpanzee                    |
| <b>monkey</b>       | marmoset, titi                          | spider monkey, howler monkey           |

Table 5: Superclasses used for the LIVING-17 task, along with the corresponding subpopulations that comprise the source and target domains.

| Superclass                      | Source                         | Target                           |
|---------------------------------|--------------------------------|----------------------------------|
| <b>bag</b>                      | plastic bag, purse             | mailbag, backpack                |
| <b>ball</b>                     | volleyball, punching bag       | ping-pong ball, soccer ball      |
| <b>boat</b>                     | gondola, trimaran              | catamaran, canoe                 |
| <b>body armor</b>               | bulletproof vest, breastplate  | chain mail, cuirass              |
| <b>bottle</b>                   | pop bottle, beer bottle        | wine bottle, water bottle        |
| <b>bus</b>                      | trolleybus, minibus            | school bus, recreational vehicle |
| <b>car</b>                      | racer, Model T                 | police van, ambulance            |
| <b>chair</b>                    | folding chair, throne          | rocking chair, barber chair      |
| <b>coat</b>                     | lab coat, fur coat             | kimono, vestment                 |
| <b>digital computer</b>         | laptop, desktop computer       | notebook, hand-held computer     |
| <b>dwelling</b>                 | palace, monastery              | mobile home, yurt                |
| <b>fence</b>                    | worm fence, chainlink fence    | stone wall, picket fence         |
| <b>hat</b>                      | bearskin, bonnet               | sombrero, cowboy hat             |
| <b>keyboard instrument</b>      | grand piano, organ             | upright, accordion               |
| <b>mercantile establishment</b> | butcher shop, barbershop       | shoe shop, grocery store         |
| <b>outbuilding</b>              | greenhouse, apiary             | barn, boathouse                  |
| <b>percussion instrument</b>    | steel drum, marimba            | drum, gong                       |
| <b>pot</b>                      | teapot, Dutch oven             | coffeepot, caldron               |
| <b>roof</b>                     | dome, vault                    | thatch, tile roof                |
| <b>ship</b>                     | schooner, pirate               | aircraft carrier, liner          |
| <b>skirt</b>                    | hoop skirt, miniskirt          | overskirt, sarong                |
| <b>stringed instrument</b>      | electric guitar, banjo         | violin, acoustic guitar          |
| <b>timepiece</b>                | digital watch, stopwatch       | parking meter, digital clock     |
| <b>truck</b>                    | fire engine, pickup            | tractor, forklift                |
| <b>wind instrument</b>          | oboe, sax                      | flute, bassoon                   |
| <b>squash</b>                   | spaghetti squash, acorn squash | zucchini, butternut squash       |

Table 6: Superclasses used for the NON-LIVING-26 task, along with the corresponding subpopulations that comprise the source and target domains.

## A.5 Annotator task

As described in Section 4.3, the goal of our human studies is to understand whether humans can classify images into superclasses even without knowing the semantic grouping. Thus, the task involved showing annotators two groups of images, each sampled from the source domain of a random superclass. Then, annotators were shown a new set of images from the target domain (or the source domain in the case of control) and were asked to assign each of them into one of the two groups. A screenshot of an (random) instance of our annotator task is shown in Figure 7.

Each task contained 20 images from the source domain of each superclass and 12 images for annotators to classify (the images were rescaled and center-cropped to size  $224 \times 224$  to match the input size used for model predictions). The two superclasses were randomly permuted at load time. To ensure good concentration of our accuracy estimates, for every superclass, we performed binary classification tasks w.r.t. 3 other (randomly chosen) superclasses. Further, we used 3 annotators per task. and annotators were compensated \$0.15 per task.

**Comparing with the original hierarchy.** In order to compare our superclasses with those obtained by Huh, Agrawal, and Efros [HAE16] via WordNet clustering,<sup>6</sup> we need to define a correspondence between them. To do so, for each of our tasks, we selected the clustering (either top-down or bottom-up) that had the closest number of superclasses. Following the terminology from that work, this mapping is: ENTITY-13 → DOWNUP-36, ENTITY-30 → UPDOWN-127, LIVING-17 → DOWNUP-753 (restricted to “living” nodes), and NON-LIVING-26 → DOWNUP-345 (restricted to “non-living” nodes).

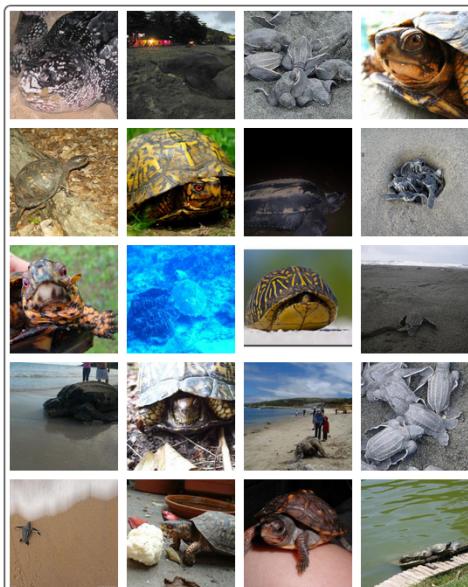
---

<sup>6</sup>[https://github.com/minyoungg/wmigftl/tree/master/label\\_sets/hierarchy](https://github.com/minyoungg/wmigftl/tree/master/label_sets/hierarchy)

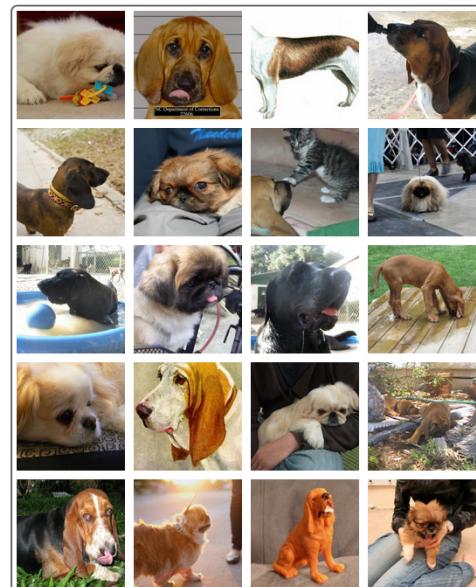
### Classify images into one of the following groups

Below you are shown example images from two groups. Your task will be to look at new images and determine the group each of them belongs to.

Group 1



Group 2



Determine the group each image belongs to

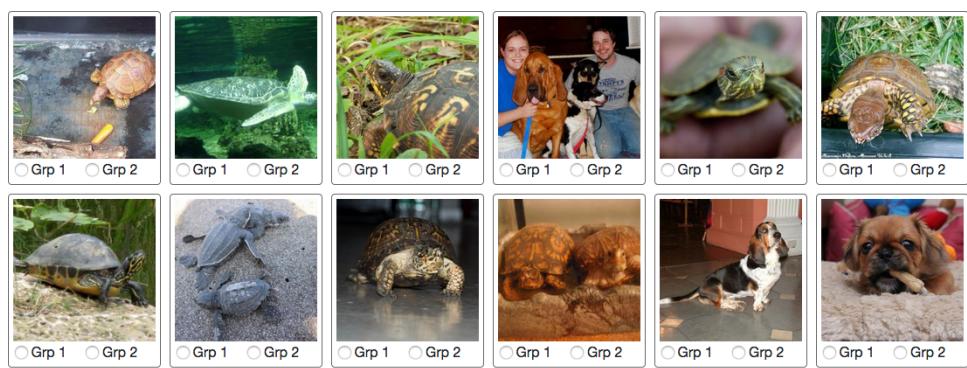


Figure 7: Sample MTurk annotation task to obtain human baselines for BREEDS benchmarks.

## A.6 Evaluating model performance

### A.6.1 Model architectures and training

The model architectures used in our analysis are in Table 7 for which we used standard implementations from the PyTorch library (<https://pytorch.org/docs/stable/torchvision/models.html>). For training, we use a batch size of 128, weight decay of  $10^{-4}$ , and learning rates listed in Table 7. Models were trained until convergence. On ENTITY-13 and ENTITY-30, this required a total of 300 epochs, with 10-fold drops in learning rate every 100 epochs, while on LIVING-17 and NON-LIVING-26, models a total of 450 epochs, with 10-fold learning rate drops every 150 epochs. For adapting models, we retrained the last (fully-connected) layer on the train split of the target domain, starting from the parameters of the source-trained model. We trained that layer using SGD with a batch size of 128 for 40,000 steps and chose the best learning rate out of [0.01, 0.1, 0.25, 0.5, 1.0, 2.0, 3.0, 5.0, 7.0, 8.0, 10.0, 11.0, 12.0], based on test accuracy.

| Model       | Learning Rate |
|-------------|---------------|
| alexnet     | 0.01          |
| vgg11       | 0.01          |
| resnet18    | 0.1           |
| resnet34    | 0.1           |
| resnet50    | 0.1           |
| densenet121 | 0.1           |

Table 7: Models used in our analysis.

### A.6.2 Model pairwise accuracy

In order to make a fair comparison between the performance of models and human annotators on the BREEDS tasks, we evaluate model accuracy on pairs of superclasses. On images from that pair, we determine the model prediction to be the superclass for which the model’s predicted probability is higher. A prediction is deemed correct if it matches the superclass label for the image. Repeating this process over random pairs of superclasses allows us to estimate model accuracy on the average-case binary classification task.

### A.6.3 Robustness interventions

For model training, we use the hyperparameters provided in Appendix A.6.1. Additional intervention-specific hyperparameters are listed in Appendix Table 8. Due to computational constraints, we trained a restricted set of model architectures with robustness interventions—ResNet-18 and ResNet-50 for adversarial training, and ResNet-18 and ResNet-34 for all others. Adversarial training was implemented using the robustness library,<sup>7</sup> while random erasing using the PyTorch transforms.<sup>8</sup>

| Eps | Step size | #Steps | Mean | StdDev | Probability | Scale       | Ratio     |
|-----|-----------|--------|------|--------|-------------|-------------|-----------|
| 0.5 | 0.4       | 3      | 0    | 0.2    | 0.5         | 0.02 - 0.33 | 0.3 - 3.3 |
| 1   | 0.8       | 3      |      |        |             |             |           |

(a) PGD-training [Mad+18]      (b) Gaussian noise      (c) Random erasing

Table 8: Additional hyperparameters for robustness interventions.

<sup>7</sup><https://github.com/MadryLab/robustness>

<sup>8</sup><https://pytorch.org/docs/stable/torchvision/transforms.html>

## B Additional Experimental Results

### B.1 Human Baselines for BREEDS Tasks

In Section 4.3, we evaluate human performance on binary versions of our BREEDS tasks. Appendix Figures 8a and 8b show the distribution of annotator accuracy over different pairs of superclasses for test data sampled from the source and target domains respectively.

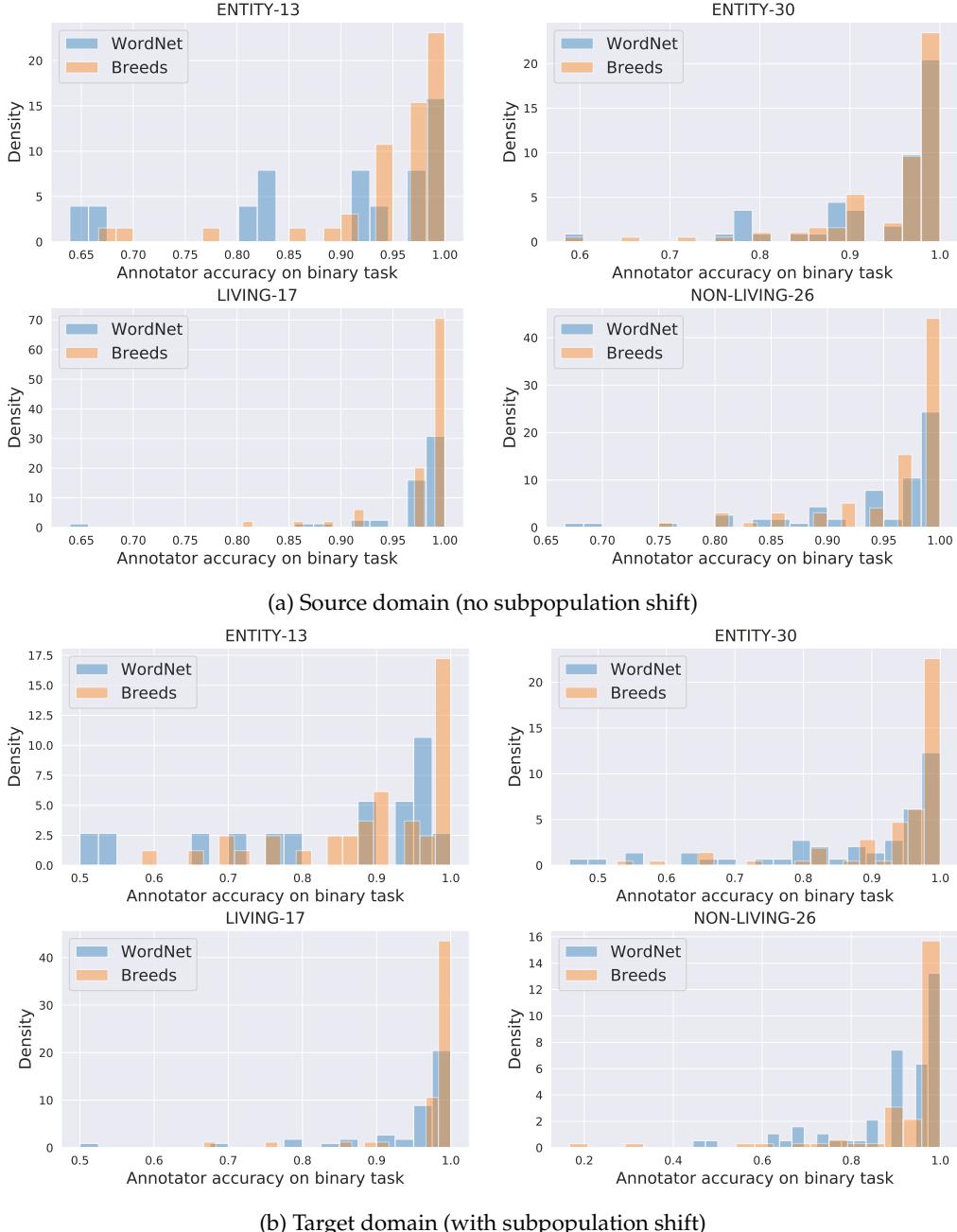


Figure 8: Distribution of annotator accuracy over pairwise superclass classification tasks. We observe that human annotators consistently perform better on tasks constructed using our modified ImageNet class hierarchy (i.e., BREEDS) as opposed to those obtained directly from WordNet.

## B.2 Model Evaluation

In Figures 9- 11, we visualize model performance over BREEDS superclasses for different model architectures. We observe in general that models perform fairly uniformly over classes when the test data is drawn from the source domain. This indicates that the tasks are well-calibrated—the various superclasses are of comparable difficulty. At the same time, we see that model robustness to subpopulation shift, i.e., drop in accuracy on the target domain, varies widely over superclasses. This could be either due to some superclasses being broader by construction or due to models being more sensitive to subpopulation shift for some classes.



Figure 9: Per-class source and target accuracies for AlexNet on BREEDS tasks.



Figure 10: Per-class source and target accuracies for ResNet-50 on BREEDS tasks.



Figure 11: Per-class source and target accuracies for DenseNet-121 on BREEDS tasks.

### B.2.1 Effect of different splits

As described in Section 3, to create BREEDS tasks, we first identify a set of relevant superclasses (at the chosen depth in the hierarchy), and then partition their subpopulations between the source and target domains. For all the tasks listed in Table 1, the superclasses are balanced—each of them comprise the same number of subpopulations. To ensure this is the case, the desired number of subpopulations is chosen among all superclass subpopulations at random. These subpopulations are then randomly split between the source and target domains.

Instead of randomly partitioning subpopulations (of a given superclass) between the two domains, we could instead craft partitions to be more/less adversarial as illustrated in Figure 12. Specifically, we could control how similar the subpopulations in the target domain are to those in the source domain. For instance, a split would be less adversarial (*good*) if subpopulations in the source and target domain share a common parent. On the other hand, we could make a split more adversarial (*bad*) by ensuring a greater degree of separation (in terms of distance in the hierarchy) between the source and target domain subpopulations.

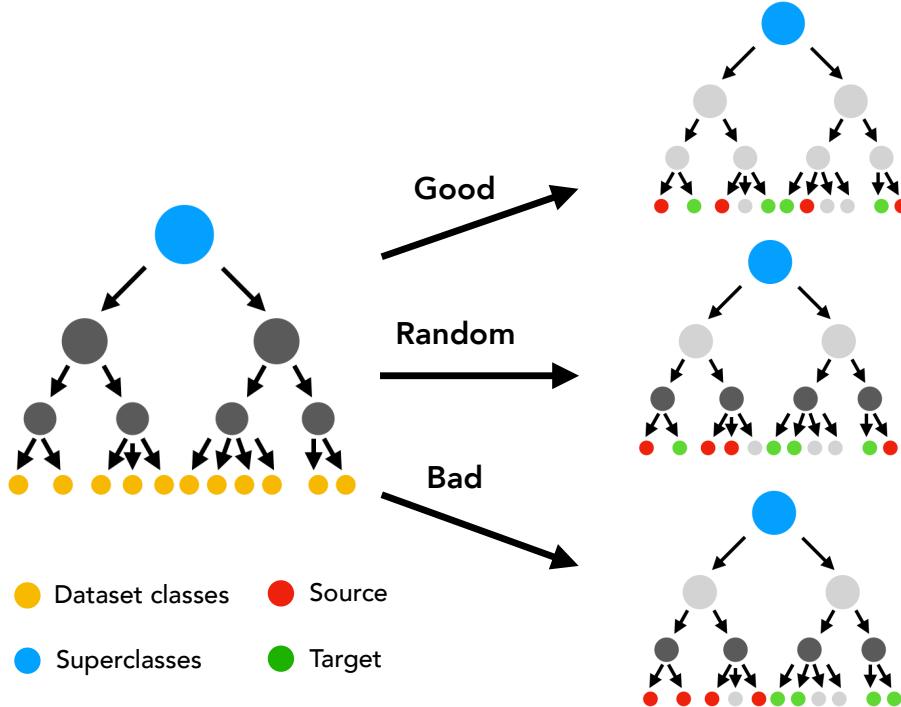
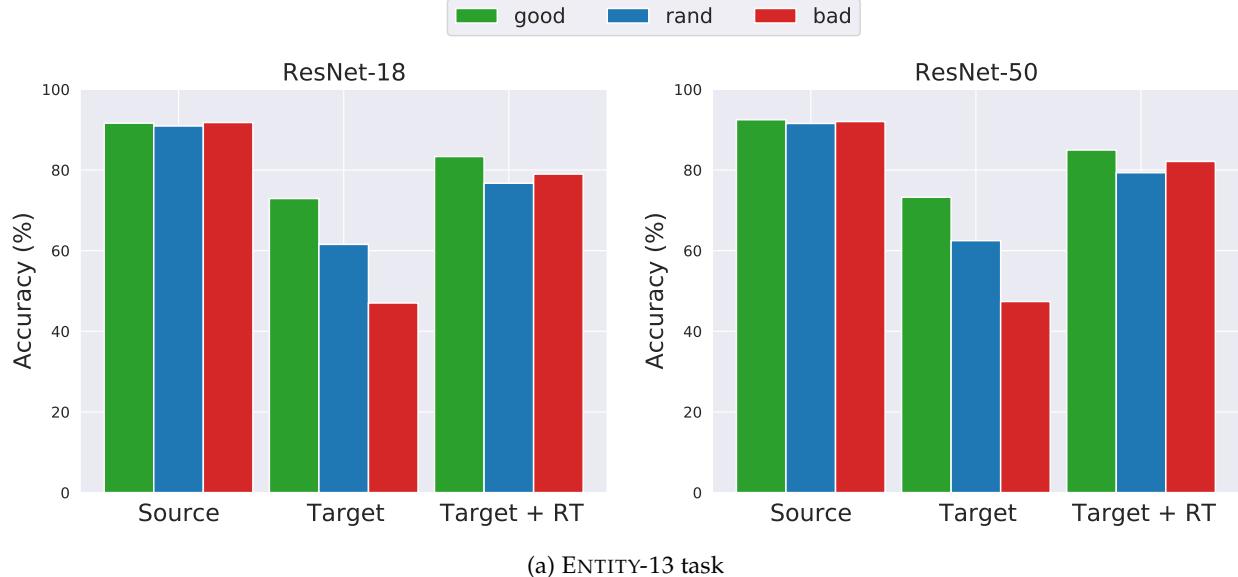


Figure 12: Different ways to partition the subpopulations of a given superclass into the source and target domains. Depending on how closely related the subpopulations in the two domain are, we can construct splits that are more/less adversarial.

We now evaluate model performance under such variations in the nature of the splits themselves—see Figure 13. As expected, models perform comparably well on test data from the source domain, independent of the how the subpopulations are partitioned into the two domains. However, model robustness to subpopulation shift varies considerably based on the nature of the split—it is lowest for the most adversarially chosen split. Finally, we observe that retraining the linear layer on data from the target domain recovers a considerable fraction of the accuracy drop in all cases—indicating that even for the more adversarial splits, models do learn features that transfer well to unknown subpopulations.



(a) ENTITY-13 task



(b) ENTITY-30 task

Figure 13: Model robustness as a function of the nature of subpopulation shift within specific BREEDS tasks. We vary how the underlying subpopulations of each superclass are split between the source and target domain—we compare random splits (used in the majority of our analysis), to ones that are more (*bad*) or less adversarial (*good*). When models are tested on samples from the source domain, they perform equally well across different splits, as one might expect. However, under subpopulation shift (i.e., on samples from the target domain), model robustness varies drastically, and is considerably worse when the split is more adversarial. Yet, for all the splits, models have comparable target accuracy after retraining their final layer.

### B.2.2 Robustness Interventions

In Tables 9 and 10, we present the raw accuracies of models trained using various train-time robustness interventions.

| ResNet-18     |            |                     |                     |                     |  |
|---------------|------------|---------------------|---------------------|---------------------|--|
| Task          | $\epsilon$ | Source              | Accuracy (%)        |                     |  |
|               |            |                     | Target              | Target-RT           |  |
| ENTITY-13     | 0          | <b>90.91 ± 0.73</b> | <b>61.52 ± 1.23</b> | <b>76.71 ± 1.09</b> |  |
|               | 0.5        | 89.23 ± 0.80        | <b>61.10 ± 1.23</b> | 74.92 ± 1.04        |  |
|               | 1.0        | 88.45 ± 0.81        | 58.53 ± 1.26        | 73.35 ± 1.11        |  |
| ENTITY-30     | 0          | <b>87.88 ± 0.89</b> | <b>49.96 ± 1.31</b> | <b>73.05 ± 1.17</b> |  |
|               | 0.5        | 85.68 ± 0.91        | <b>48.93 ± 1.34</b> | 71.34 ± 1.14        |  |
|               | 1.0        | 84.23 ± 0.91        | 47.66 ± 1.23        | 70.27 ± 1.17        |  |
| LIVING-17     | 0          | <b>92.01 ± 1.30</b> | <b>58.21 ± 2.32</b> | <b>83.38 ± 1.79</b> |  |
|               | 0.5        | 90.35 ± 1.35        | 55.79 ± 2.44        | <b>83.00 ± 1.89</b> |  |
|               | 1.0        | 88.56 ± 1.50        | 53.89 ± 2.36        | 80.90 ± 1.92        |  |
| NON-LIVING-26 | 0          | <b>88.09 ± 1.28</b> | <b>41.87 ± 2.01</b> | <b>73.52 ± 1.71</b> |  |
|               | 0.5        | 86.28 ± 1.32        | <b>41.02 ± 1.91</b> | <b>72.41 ± 1.71</b> |  |
|               | 1.0        | 85.19 ± 1.38        | <b>40.23 ± 1.92</b> | 70.61 ± 1.73        |  |

| ResNet-50     |            |                     |                     |                     |  |
|---------------|------------|---------------------|---------------------|---------------------|--|
| Task          | $\epsilon$ | Source              | Accuracy (%)        |                     |  |
|               |            |                     | Target              | Target-RT           |  |
| ENTITY-13     | 0          | <b>91.54 ± 0.64</b> | <b>62.48 ± 1.16</b> | <b>79.32 ± 1.01</b> |  |
|               | 0.5        | 89.87 ± 0.80        | <b>63.01 ± 1.15</b> | <b>80.14 ± 1.00</b> |  |
|               | 1.0        | 89.71 ± 0.74        | 61.21 ± 1.22        | 78.58 ± 0.98        |  |
| ENTITY-30     | 0          | <b>89.26 ± 0.78</b> | <b>51.18 ± 1.24</b> | 77.60 ± 1.17        |  |
|               | 0.5        | 87.51 ± 0.88        | <b>50.72 ± 1.28</b> | <b>78.92 ± 1.06</b> |  |
|               | 1.0        | 86.63 ± 0.88        | <b>50.99 ± 1.27</b> | <b>78.63 ± 1.03</b> |  |
| LIVING-17     | 0          | <b>92.40 ± 1.28</b> | <b>58.22 ± 2.42</b> | <b>85.96 ± 1.72</b> |  |
|               | 0.5        | 90.79 ± 1.55        | <b>55.97 ± 2.38</b> | <b>87.22 ± 1.66</b> |  |
|               | 1.0        | 89.64 ± 1.47        | 54.64 ± 2.48        | <b>85.63 ± 1.73</b> |  |
| NON-LIVING-26 | 0          | <b>88.13 ± 1.30</b> | <b>41.82 ± 1.86</b> | 76.58 ± 1.69        |  |
|               | 0.5        | <b>88.20 ± 1.20</b> | <b>42.57 ± 2.03</b> | <b>78.84 ± 1.62</b> |  |
|               | 1.0        | 86.17 ± 1.36        | <b>41.69 ± 1.96</b> | 76.16 ± 1.61        |  |

Table 9: Effect of adversarial training on model robustness to subpopulation shift. All models are trained on samples from the source domain—either using standard training ( $\epsilon = 0.0$ ) or using adversarial training. Models are then evaluated in terms of: (a) source accuracy, (b) target accuracy and (c) target accuracy after retraining the linear layer of the model with data from the target domain. Confidence intervals (95%) obtained via bootstrapping. Maximum task accuracy over  $\epsilon$  (taking into account confidence interval) shown in bold.

| ResNet-18     |                   |                     |                     |                     |           |
|---------------|-------------------|---------------------|---------------------|---------------------|-----------|
| Task          | Intervention      | Accuracy (%)        |                     |                     | Target-RT |
|               |                   | Source              | Target              |                     |           |
| ENTITY-13     | Standard          | <b>90.91 ± 0.73</b> | <b>61.52 ± 1.23</b> | 76.71 ± 1.09        |           |
|               | Erase Noise       | <b>91.01 ± 0.68</b> | <b>62.79 ± 1.27</b> | <b>78.10 ± 1.09</b> |           |
|               | Gaussian Noise    | 77.00 ± 1.04        | 47.90 ± 1.21        | 70.37 ± 1.17        |           |
|               | Stylized ImageNet | 76.85 ± 1.00        | 50.18 ± 1.21        | 65.91 ± 1.17        |           |
| ENTITY-30     | Standard          | <b>87.88 ± 0.89</b> | <b>49.96 ± 1.31</b> | 73.05 ± 1.17        |           |
|               | Erase Noise       | <b>88.09 ± 0.80</b> | <b>49.98 ± 1.31</b> | <b>74.27 ± 1.15</b> |           |
|               | Gaussian Noise    | 74.12 ± 1.16        | 35.79 ± 1.21        | 65.62 ± 1.28        |           |
|               | Stylized ImageNet | 70.96 ± 1.16        | 37.67 ± 1.21        | 60.45 ± 1.22        |           |
| LIVING-17     | Standard          | <b>92.01 ± 1.30</b> | <b>58.21 ± 2.32</b> | <b>83.38 ± 1.79</b> |           |
|               | Erase Noise       | <b>93.09 ± 1.27</b> | <b>59.60 ± 2.40</b> | <b>85.12 ± 1.71</b> |           |
|               | Gaussian Noise    | 80.13 ± 1.99        | 46.16 ± 2.57        | 77.31 ± 2.08        |           |
|               | Stylized ImageNet | 79.21 ± 1.85        | 43.96 ± 2.38        | 72.74 ± 2.09        |           |
| NON-LIVING-26 | Standard          | <b>88.09 ± 1.28</b> | <b>41.87 ± 2.01</b> | <b>73.52 ± 1.71</b> |           |
|               | Erase Noise       | <b>88.68 ± 1.18</b> | <b>43.17 ± 2.10</b> | <b>73.91 ± 1.78</b> |           |
|               | Gaussian Noise    | 78.14 ± 1.60        | 35.13 ± 1.94        | 67.79 ± 1.79        |           |
|               | Stylized ImageNet | 71.43 ± 1.73        | 30.56 ± 1.75        | 61.83 ± 1.98        |           |
| ResNet-34     |                   |                     |                     |                     |           |
| Task          | Intervention      | Accuracy (%)        |                     |                     | Target-RT |
|               |                   | Source              | Target              |                     |           |
| ENTITY-13     | Standard          | <b>91.75 ± 0.70</b> | <b>63.45 ± 1.13</b> | <b>78.07 ± 1.02</b> |           |
|               | Erase Noise       | <b>91.76 ± 0.70</b> | <b>62.71 ± 1.25</b> | <b>77.43 ± 1.06</b> |           |
|               | Gaussian Noise    | 81.60 ± 0.97        | 50.69 ± 1.28        | 71.50 ± 1.13        |           |
|               | Stylized ImageNet | 78.66 ± 0.94        | 51.05 ± 1.30        | 67.38 ± 1.16        |           |
| ENTITY-30     | Standard          | <b>88.81 ± 0.81</b> | <b>51.68 ± 1.28</b> | <b>75.12 ± 1.11</b> |           |
|               | Erase Noise       | <b>89.07 ± 0.82</b> | <b>51.04 ± 1.27</b> | <b>74.88 ± 1.08</b> |           |
|               | Gaussian Noise    | 75.05 ± 1.11        | 38.31 ± 1.26        | 67.47 ± 1.22        |           |
|               | Stylized ImageNet | 72.51 ± 1.10        | 38.98 ± 1.22        | 61.65 ± 1.25        |           |
| LIVING-17     | Standard          | <b>92.83 ± 1.19</b> | 59.74 ± 2.27        | <b>85.46 ± 1.83</b> |           |
|               | Erase Noise       | <b>92.96 ± 1.32</b> | <b>61.13 ± 2.30</b> | <b>85.66 ± 1.78</b> |           |
|               | Gaussian Noise    | 84.06 ± 1.71        | 48.38 ± 2.44        | 78.79 ± 1.91        |           |
|               | Stylized ImageNet | 80.94 ± 2.00        | 44.16 ± 2.43        | 72.77 ± 2.18        |           |
| NON-LIVING-26 | Standard          | <b>89.64 ± 1.17</b> | <b>43.03 ± 1.99</b> | <b>74.99 ± 1.66</b> |           |
|               | Erase Noise       | <b>89.62 ± 1.31</b> | <b>43.53 ± 1.89</b> | <b>75.04 ± 1.70</b> |           |
|               | Gaussian Noise    | 79.26 ± 1.61        | 34.89 ± 1.91        | 68.07 ± 1.78        |           |
|               | Stylized ImageNet | 71.49 ± 1.65        | 31.10 ± 1.80        | 62.94 ± 1.90        |           |

Table 10: Effect of various train-time interventions on model robustness to subpopulation shift. All models are trained on samples from the source domain. Models are then evaluated in terms of: (a) source accuracy, (b) target accuracy and (c) target accuracy after retraining the linear layer of the model with data from the target domain. Confidence intervals (95%) obtained via bootstrapping. Maximum task accuracy over  $\varepsilon$  (taking into account confidence interval) shown in bold.