Ask Your Distribution Shift if Pre-Training is Right for You

Benjamin Cohen-Wang
MIT
bencw@mit.edu

Joshua Vendrow
MIT
jvendrow@mit.edu

Aleksander Mądry
MIT
madry@mit.edu

Abstract

Pre-training is a widely used approach to develop models that are robust to distribution shifts. However, in practice, its effectiveness varies: fine-tuning a pre-trained model improves robustness significantly in some cases but *not at all* in others (compared to training from scratch). In this work, we seek to characterize the failure modes that pre-training *can* and *cannot* address. In particular, we focus on two possible failure modes of models under distribution shift: poor extrapolation (e.g., they cannot generalize to a different domain) and biases in the training data (e.g., they rely on spurious features). Our study suggests that, as a rule of thumb, pre-training can help mitigate poor extrapolation but not dataset biases. After providing theoretical motivation and empirical evidence for this finding, we explore an implication for developing robust models: fine-tuning on a (very) small, non-diverse but *de-biased* dataset can result in significantly more robust models than fine-tuning on a large and diverse but biased dataset.

1 Introduction

A common paradigm for developing machine learning models is pre-training them on a large, diverse dataset (e.g., ImageNet [1], JFT-300M [2], LAION-5B [3]) and then fine-tuning them on task-specific data. Indeed, compared to training from scratch, fine-tuning a pre-trained model often significantly improves performance and reduces computational costs [4, 2, 5].

Yet another benefit that pre-training may offer is *distribution shift robustness*. Specifically, machine learning models tend to suffer from distribution shifts, i.e., changes between the *reference distribution* used to develop the model and the *shifted distribution* that the model actually encounters when deployed. For example, a tumor identification model trained on tissue slide images from one hospital might perform poorly when deployed at another hospital [6, 7]. Notably, different models (with different architectures, hyperparameters, etc.) tend to be similarly sensitive to a given distribution shift. However, models pre-trained on auxiliary data and then fine-tuned on the reference distribution can break this trend, exhibiting substantially higher performance on the shifted distribution than models trained from scratch with the same performance on the reference distribution [8–10].

These robustness benefits of pre-training are promising, but they are *not* universal. In particular, fine-tuning the same pre-trained model can yield significant robustness gains on some distribution shifts but not on others [11, 12]. To attain robustness to the latter shifts, would fine-tuning a larger model pre-trained on more data suffice? Or are there fundamental limitations to the robustness that pre-training can provide? To answer these questions, we develop a more fine-grained understanding of how pre-training improves robustness. Specifically, we ask:

Can we identify and characterize the failure modes that pre-training <u>can</u> and <u>cannot</u> address?

Recall that under distribution shift, models can fail in a number of ways. One of them is their inability to *extrapolate* effectively outside of the reference distribution [13, 7]. If, for instance, a model is trained only on photos taken during the day, then it may fail when deployed on photos taken at night.

Workshop on Distribution Shifts, 37th Conference on Neural Information Processing Systems (NeurIPS 2023).

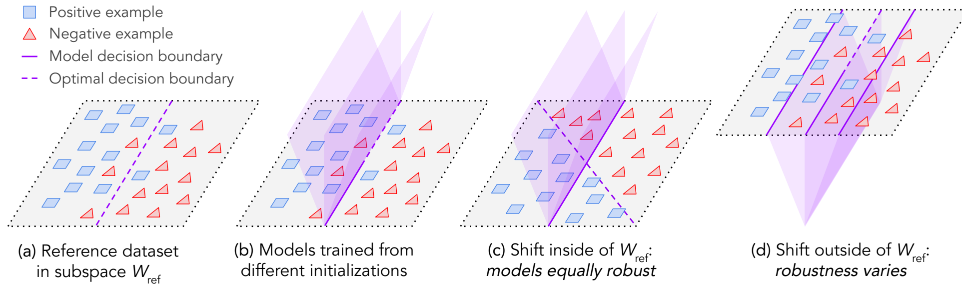


Figure 1: **Illustration of logistic regression setting.** (a) Consider a reference dataset that lies within a subspace W_{ref} of $\mathbb{R}^d$. (b) Models trained from different initializations all learn the same (optimal) decision boundary in W_{ref}, but may behave differently outside of W_{ref}. (c) Within W_{ref}, models with different initializations are equally robust. (d) Outside of W_{ref}, initialization can affect robustness.

Models can also underperform even when the shifted distribution does not contain anything "new." In particular, they can fail due to *biases* in the reference distribution. For example, if a certain feature is spuriously correlated with the label in the reference distribution, a model might learn to exploit this relationship and fail on examples encountered during deployment where it does not hold [14, 15].

Our contributions. To identify the failure modes that pre-training can address, we study the robustness benefits of pre-training under two types of distribution shifts: (1) shifts where extrapolation is necessary and (2) shifts where extrapolation is not needed. We start by analyzing a simple logistic regression setting and illustrate why pre-training might improve robustness to the former type of shift, but not the latter (Section 3). We subsequently build on this intuition by measuring the robustness benefits of pre-training on shifts of each type (Section 4). Our results suggest the following rule of thumb: pre-training helps specifically with extrapolation, but does not address other failures, for example, those stemming from dataset biases.

Guided by this rule of thumb, we explore one potential avenue for harnessing pre-training to develop robust models: curating a small, non-diverse but *de-biased* fine-tuning dataset (Section 5). Specifically, we demonstrate that fine-tuning on a hair color classification dataset with only 64 examples that was carefully de-biased yields greater robustness than fine-tuning on the entire CelebA dataset [16]. We provide related work in Appendix A, a conclusion in Appendix B and discussion in Appendix C.

2 Background

Fine-tuning a pre-trained model. Two common strategies for fine-tuning a pre-trained model are *full fine-tuning*, in which one continues training the entire model, and *linear probing*, in which one only fine-tunes the final layer. Some recent pre-trained models with natural language supervision (e.g., CLIP [11], ALIGN [17]) can also be applied to a downstream task in a *zero-shot* context (i.e., without fine-tuning) by specifying the task through a text description. In this work, we focus on full fine-tuning (and sometimes initialize using linear probing [18] or zero-shot adaptation [11]).

Measuring robustness. For many distribution shifts, different models trained from scratch on the reference distribution exhibit similar degrees of robustness to the shift. Specifically, when varying architectures, hyperparameters and training methods there is often a strong *linear* relationship between the reference accuracy and shifted accuracy[1] [8–10]. This relationship, dubbed *accuracy on the line*, can be visualized by plotting shifted accuracies against reference accuracies and finding a linear fit. To quantify the robustness of a model trained with a robustness intervention beyond the "baseline" of models trained from scratch, one can measure the amount by which its shifted accuracy exceeds the linear fit's prediction, a metric known as *effective robustness* (ER) [8] (see, e.g., Figure 2). See Appendix F.1.2 for additional details.

3 Studying Pre-Training in a Logistic Regression Setting

Our central goal is to understand the failure modes that pre-training *can* and *cannot* address. To this end, we first study its robustness benefits in a simple logistic regression setting (see Figure 1).

Setup. We are given access to a reference dataset S_{ref} of input-label pairs, each consisting of a d-dimensional input $x \in \mathbb{R}^d$ and a binary label $y \in \{-1, 1\}$. In particular, we assume that inputs in the reference dataset S_{ref} lie within a k-dimensional (with $k < d$) subspace W_{ref} of $\mathbb{R}^d$. Intuitively, this condition corresponds to features lacking certain variation in the reference dataset.

To understand how a (pre-trained or randomly initialized) model trained on S_{ref} performs under distribution shift, we establish the following theorem (exact setup and proof in Appendix D):

Theorem 3.1. *Suppose that we start with initial weights $w_{init} \in \mathbb{R}^d$ and run gradient descent to minimize the logistic loss $L_{ref}(w)$ on S_{ref}. With an appropriately chosen learning rate, gradient descent converges to weights $\hat{w}$ that minimize L_{ref}. Furthermore, $\hat{w}$ can be written as*

$$\hat{w} = w^*_{ref} + (w_{init} - proj_{W_{ref}} w_{init}) \tag{1}$$

where w^*_{ref} does not depend on the initialization w_{init} and lies within the reference subspace W_{ref}.

Theorem 3.1 enables us to decompose the learned model's weights $\hat{w}$ into two terms: w^*_{ref} and $(w_{\text{init}} - \text{proj}_{W_{\text{ref}}} w_{\text{init}})$. Notice that the first term is just a property of the reference dataset and is in the reference subspace W_{ref}, while the second term depends on w_{init} and is *orthogonal* to W_{ref}. As a result, the reference dataset itself fully specifies the model's behavior on inputs in W_{ref}, while the initialization determines how the model extends outside of W_{ref}.

This observation suggests the following key intuition: pre-training can improve robustness to a distribution shift *only* when the shifted distribution contains "out-of-support" inputs, that is, inputs that could not be reasonably sampled from the reference distribution. In other words, pre-training helps specifically with extrapolation outside of the reference distribution.

4 Exploring the Empirical Robustness Benefits of Pre-Training

In Section 3, we found that in a simple logistic regression setting, pre-training helps *specifically* with extrapolation. We now want to assess whether this principle holds more broadly. To do so, we measure the robustness benefits of pre-training under two types of shifts.

In-support shift. A distribution shift is *in-support* if any input that could be sampled from the shifted distribution could also be reasonably sampled from the reference distribution. Under an in-support shift, the shifted distribution does not contain anything "new" so models *cannot* fail due to poor extrapolation (but might fail for other reasons, e.g., dataset biases).

Out-of-support shift. A distribution shift is *out-of-support* if there exists an input that could be sampled from the shifted distribution but could not be reasonably sampled from the reference distribution. Hence, models *can* fail due to poor extrapolation.

Evaluating robustness to in-support and out-of-support shifts. We now want to evaluate the robustness gains that pre-training provides on in-support and out-of-support shifts. To this end, we explicity construct two shifts of each type by modifying CIFAR-10 [19] (see the top of Figure 2 for visualizations). For each shift, as a baseline, we train ResNet-18 [20] models from scratch on differently sized subsets of the reference dataset. Next, we fine-tune various pre-trained models and measure their effective robustness above this baseline. See Appendix F.2 for the exact setup.

We observe that pre-trained models exhibit substantial effective robustness on out-of-support shifts, but have very limited (yet non-zero) effective robustness on in-support shifts (see Figure 2). This broadly supports our hypothesis. However, note that if pre-training indeed only helped with extrapolation, then we might initially expect that it would yield *no* effective robustness to in-support shifts at all. We investigate this limited effective robustness of pre-trained models on in-support shifts in Appendix E.1.2 and conclude that it likely also derives from better extrapolation. In Appendix E.1.3, we show that these findings seem to extend to natural distribution shifts as well.

[1] For a linear relationship, accuracies are *probit-scaled* (transformed by the inverse of the Gaussian CDF).

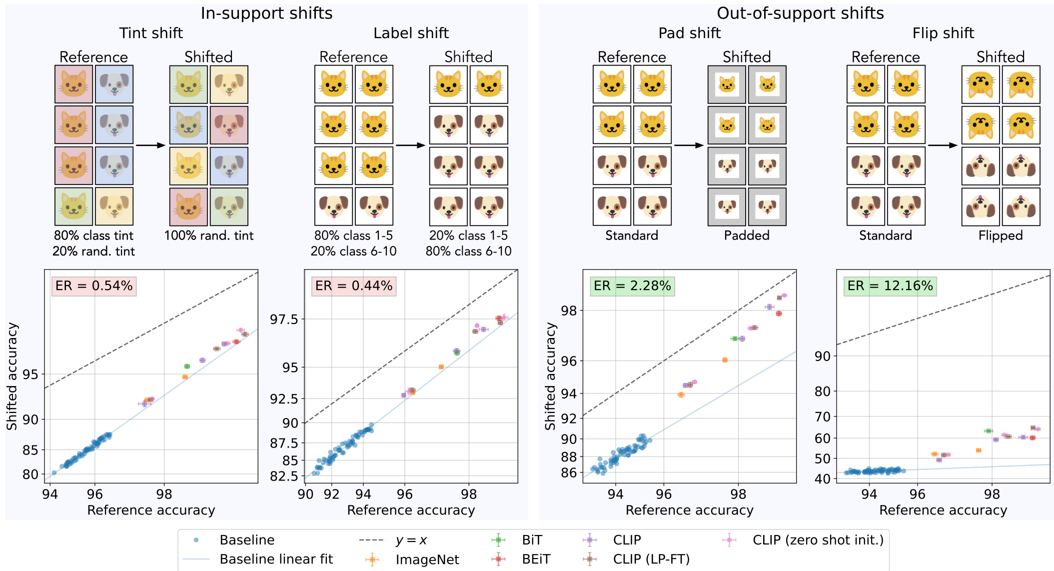
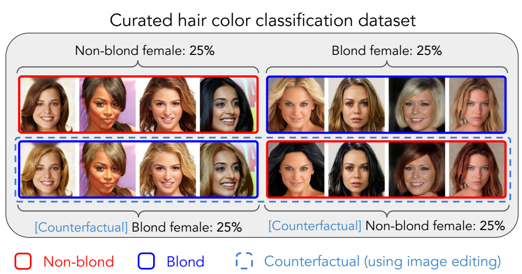
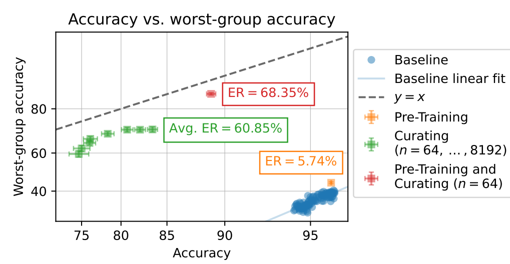


Figure 2: **Robustness of pre-trained models to in-support and out-of-support shifts.** For each of two in-support shifts (left) and two out-of-support shifts (right) constructed by modifying CIFAR-10, the reference and shifted accuracies of models trained from scratch (in blue) are linearly correlated. Pre-trained models exhibit little effective robustness (i.e., little improvement over the linear trend, see Section 2) on the in-support shifts, but have significant effective robustness on the out-of-support shifts (averages in the top left of each plot).



(a) In our curated dataset, every real image (top row) is paired with a synthesized "counterfactual example" of the other class (bottom row). As a result, the primary difference between the blond and non-blond populations is hair color; other attributes such as gender, age and hair style are not predictive. We include only females in our dataset to illustrate that diversity might not be necessary for robustness when fine-tuning.



(b) Fine-tuning a pre-trained model on the CelebA dataset (orange) yields little effective robustness (ER) over models trained from scratch (blue). However, fine-tuning the same pre-trained model on just 64 curated examples (red) yields a model with high ER *and* accuracy. Training from scratch on our curated dataset (green) also yields high ER, but results in substantially lower accuracy, even with many more examples.

Figure 3: Fine-tuning a pre-trained model on a small, non-diverse but de-biased dataset (see Figure 3a) yields a robust and performant model for hair color classification in CelebA (see Figure 3b).

5 Curating Datasets for Fine-Tuning

Our observations in Section 4 suggest that pre-training indeed helps prevent failures caused by poor extrapolation but not those stemming from dataset biases. Guided by this rule of thumb, we highlight one strategy for developing robust models: fine-tuning a pre-trained model on a carefully curated (and, in particular, de-biased) dataset *instead* of the original reference dataset. If we can rely on pre-training for extrapolation, we might only need a small, non-diverse fine-tuning dataset, which could be feasible to de-bias. Thus, curating such a dataset and then fine-tuning a large pre-trained model on it might be a relatively inexpensive method for developing robust and performant models.

As a case study, we consider the task of predicting hair color (blond vs. non-blond) in the CelebA dataset [16]. In this dataset, hair color is spuriously correlated with other attributes (especially gender). Following works studying *group robustness* [21–23], we measure worst-group accuracy to assess robustness rather than measuring accuracy on an explicit shifted dataset. In this case, the four groups are blond females, non-blond females, blond males and non-blond males.

To curate a de-biased dataset for hair color classification, we augment existing CelebA examples with synthesized "counterfactual examples" of the opposite class (see Figure 3a for a visualization and Appendix F.4 for full details). To illustrate that this dataset does *not* need to be diverse to yield high robustness and performance when fine-tuning, we restrict the dataset to include *only* females.

As expected, models trained on the CelebA dataset exhibit high accuracy but very low worst-group accuracy, regardless of whether they are pre-trained (see Figure 3b). However, we observe that fine-tuning a pre-trained model on *just* 64 examples from our curated dataset yields a model with both high accuracy *and* effective robustness. Finally, we also train models from scratch on our curated dataset and find that they also exhibit substantial effective robustness, but require many more examples to attain a comparable accuracy. This suggests that the extrapolation benefits of pre-training are key to make effective use of our small, non-diverse curated dataset.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition (CVPR)*, 2009.

[2] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, 2017.

[3] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. In *arXiv preprint arXiv:2210.08402*, 2022.

[4] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *conference on computer vision and pattern recognition (CVPR) workshops*, 2014.

[5] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *computer vision and pattern recognition (CVPR)*, 2019.

[6] Peter Bandi, Oscar Geessink, Quirine Manson, Marcory Van Dijk, Maschenka Balkenhol, Meyke Hermsen, Babak Ehteshami Bejnordi, Byungjae Lee, Kyunghyun Paeng, Aoxiao Zhong, et al. From detection of individual metastases to classification of lymph node status at the patient level: the camelyon17 challenge. *IEEE Transactions on Medical Imaging*, 2018.

[7] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Sara Beery, et al. Wilds: A benchmark of in-the-wild distribution shifts. *arXiv preprint arXiv:2012.07421*, 2020.

[8] Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[9] John Miller, Karl Krauth, Benjamin Recht, and Ludwig Schmidt. The effect of natural distribution shift on question answering models. In *International Conference on Machine Learning*, pages 6905–6916. PMLR, 2020.

[10] John P Miller, Rohan Taori, Aditi Raghunathan, Shiori Sagawa, Pang Wei Koh, Vaishaal Shankar, Percy Liang, Yair Carmon, and Ludwig Schmidt. Accuracy on the line: on the strong correlation between out-of-distribution and in-distribution generalization. In *International Conference on Machine Learning*, pages 7721–7735. PMLR, 2021.

[11] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *arXiv preprint arXiv:2103.00020*, 2021.

[12] Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo-Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. Robust fine-tuning of zero-shot models. In *arXiv preprint arXiv:2109.01903*, 2021.

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

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

[15] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. In *Nature Machine Intelligence*, 2020.

[16] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *International Conference on Computer Vision (ICCV)*, 2015.

[17] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.

[18] Ananya Kumar, Tengyu Ma, and Percy Liang. Understanding self-training for gradual domain adaptation. In *International Conference on Machine Learning (ICML)*, 2020.

[19] Alex Krizhevsky. Learning multiple layers of features from tiny images. In *Technical report*, 2009.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[21] Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *International Conference on Learning Representations*, 2020.

[22] Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, 2021.

[23] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv preprint arXiv:2204.02937*, 2022.

[24] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. *arXiv preprint arXiv:1912.11370*, 2019.

[25] Michael Xie, Neal Jean, Marshall Burke, David Lobell, and Stefano Ermon. Transfer learning from deep features for remote sensing and poverty mapping. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[26] Alexander Ke, William Ellsworth, Oishi Banerjee, Andrew Y Ng, and Pranav Rajpurkar. Chextransfer: performance and parameter efficiency of imagenet models for chest x-ray interpretation. In *Proceedings of the Conference on Health, Inference, and Learning*, pages 116–124, 2021.

[27] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4918–4927, 2019.

[28] Samira Abnar, Mostafa Dehghani, Behnam Neyshabur, and Hanie Sedghi. Exploring the limits of large scale pre-training. *arXiv preprint arXiv:2110.02095*, 2021.

[29] Hadi Salman, Saachi Jain, Andrew Ilyas, Logan Engstrom, Eric Wong, and Aleksander Madry. When does bias transfer in transfer learning? In *arXiv preprint arXiv:2207.02842*, 2022.

[30] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization, 2020.

[31] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In *International conference on machine learning*, pages 872–881. PMLR, 2019.

[32] Badr Youbi Idrissi, Martin Arjovsky, Mohammad Pezeshki, and David Lopez-Paz. Simple data balancing achieves competitive worst-group-accuracy. In *Conference on Causal Learning and Reasoning*, pages 336–351. PMLR, 2022.

[33] Karan Goel, Albert Gu, Yixuan Li, and Christopher Ré. Model patching: Closing the subgroup performance gap with data augmentation. *arXiv preprint arXiv:2008.06775*, 2020.

[34] Olivia Wiles, Sven Gowal, Florian Stimberg, Sylvestre Alvise-Rebuffi, Ira Ktena, Krishnamurthy Dvijotham, and Taylan Cemgil. A fine-grained analysis on distribution shift. *arXiv preprint arXiv:2110.11328*, 2021.

[35] Nanyang Ye, Kaican Li, Haoyue Bai, Runpeng Yu, Lanqing Hong, Fengwei Zhou, Zhenguo Li, and Jun Zhu. Ood-bench: Quantifying and understanding two dimensions of out-of-distribution generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7947–7958, 2022.

[36] Shibani Santurkar, Dimitris Tsipras, and Aleksander Madry. Breeds: Benchmarks for subpopulation shift. In *International Conference on Learning Representations (ICLR)*, 2021.

[37] Yuzhe Yang, Haoran Zhang, Dina Katabi, and Marzyeh Ghassemi. Change is hard: A closer look at subpopulation shift. *arXiv preprint arXiv:2302.12254*, 2023.

[38] Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. An investigation of why overparameterization exacerbates spurious correlations. In *International Conference on Machine Learning*, pages 8346–8356. PMLR, 2020.

[39] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 2000.

[40] Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. Detecting and correcting for label shift with black box predictors. In *International conference on machine learning*, pages 3122–3130. PMLR, 2018.

[41] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning*, pages 2712–2721. PMLR, 2019.

[42] Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. Pretrained transformers improve out-of-distribution robustness. *arXiv preprint arXiv:2004.06100*, 2020.

[43] Lifu Tu, Garima Lalwani, Spandana Gella, and He He. An empirical study on robustness to spurious correlations using pre-trained language models. *Transactions of the Association for Computational Linguistics*, 8:621–633, 2020.

[44] Anders Andreassen, Yasaman Bahri, Behnam Neyshabur, and Rebecca Roelofs. The evolution of out-of-distribution robustness throughout fine-tuning. *arXiv preprint arXiv:2106.15831*, 2021.

[45] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[46] Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054*, 2022.

[47] John Hewitt, Xiang Lisa Li, Sang Michael Xie, Benjamin Newman, and Percy Liang. Ensembles and cocktails: Robust finetuning for natural language generation. 2021.

[48] Sébastien Bubeck. Theory of convex optimization for machine learning. *arXiv preprint arXiv:1405.4980*, 15, 2014.

[49] Preetum Nakkiran, Behnam Neyshabur, and Hanie Sedghi. The deep bootstrap framework: Good online learners are good offline generalizers. *arXiv preprint arXiv:2010.08127*, 2020.

[50] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning (ICML)*, 2019.

[51] Haohan Wang, Songwei Ge, Eric P Xing, and Zachary C Lipton. Learning robust global representations by penalizing local predictive power. *Neural Information Processing Systems (NeurIPS)*, 2019.

[52] Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In *Neural Information Processing Systems (NeurIPS)*, 2019.

[53] Vaishaal Shankar, Achal Dave, Rebecca Roelofs, Deva Ramanan, Benjamin Recht, and Ludwig Schmidt. Do image classifiers generalize across time? *arXiv preprint arXiv:1906.02168*, 2019.

[54] Guillaume Leclerc, Andrew Ilyas, Logan Engstrom, Sung Min Park, Hadi Salman, and Aleksander Madry. ffcv. `https://github.com/libffcv/ffcv/`, 2022.

[55] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

[56] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 630–645. Springer, 2016.

[57] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

[58] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, 2021.

[59] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[60] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.

[61] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5549–5558, 2020.

[62] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023.

[63] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

A Related Work

Pre-training. Pre-training a model (or taking an existing pre-trained model) and then fine-tuning it on a task-specific dataset is a common practice when developing machine learning models, often significantly improving performance over training a model from scratch [4, 2, 5, 24]. Pre-training can be effective even when the downstream task is unrelated to the pre-training task, suggesting that pre-training yields useful general-purpose features; for example, object classification models trained on ImageNet [1] are good initializations for remote sensing [25] and medical imaging [26] tasks. Although greatly effective, pre-training is not without limitations. In some settings, pre-training does not improve performance over a randomly initialized model trained for long enough [27]. Downstream performance can saturate as performance on the pre-training task improves [28]. Finally, biases of pre-trained models can persist after fine-tuning [29].

Distribution shift robustness. Machine learning models are often deployed in different environments from those in which they are trained. Such distribution shifts can cause models to significantly underperform [7, 13, 30]. Numerous interventions have been proposed to improve the robustness of models, often targeting particular types of shifts. These include algorithmic interventions [14, 31, 21–23, 32] (often requiring group information), data augmentations [30, 33] and pre-training (discussed below). However, interventions proposed thus far have failed to provide consistent benefits across distribution shift benchmarks [7, 13, 30, 34, 35], rendering distribution shift robustness a persistent challenge.

Characterizing distribution shifts. There exists a plethora of definitions for characterizing distribution shifts, many of which are aligned with the in-support and out-of-support chracterizations that we discuss in this work. For example, *domain generalization* involves shifts in which the reference and shifted distributions are from different domains [7, 13]. In a *subpopulation shift*, subpopulations appear with different frequencies in the reference and shifted distributions [36, 7, 37]. In shifts with *spurious correlations*, certain features are predictive in the reference distribution but not in the shifted distribution [14, 38]. Two more formal characterizations are *covariate shift* [39], under which $p(y|x)$ is fixed, and *label shift* [40], under which the label distribution may change but $p(x|y)$ is fixed. We relate these definitons to in-support and out-of-support shifts in Appendix **??**.

The characterizations relevant in this work, *in-support shift* and *out-of-support shift*, overlap with many existing definitions. Ye et al. [35] introduce notions of *correlation shift* and *diversity shift* (closely aligned with in-support and out-of-support shifts, respectively) and provide a method for measuring the "amount" of each type of shift in a given distribution shift (similar to our method for dividing a distribution shift into in-support and out-of-support splits). Subpopulation shift (and its sub-types), shifts involving spurious correlations, covariate shift, and label shift are typically in-support. However, there are exceptions; for example, some works consider subpopulation shifts in which a subpopulation does not appear in the reference distribution [36, 37], which are out-of-support. Domain generalization problems are nearly always out-of-support and extrapolating effectively outside of the reference distribution is often a key challenge of these tasks.

Robustness benefits of pre-training. Several works have suggested that pre-training can be an effective strategy for improving robustness to distribution shifts [41, 30, 42, 43, 34, 44]. In particular, Wiles et al. [34] define different types of distribution shifts and find that pre-training frequently improves performance under these shifts, while most other interventions primarily help in specific settings. In the natural language processing setting, Tu et al. [43] argue that when pre-training helps with spurious correlations, it is because pre-trained models can generalize better from the small number of counterexamples to these correlations; as we discuss in Appendix C.3, this is consistent with our intuition that pre-training helps specifically with extrapolation. Lastly, Bommasani et al. [45] discuss failure modes that pre-training is unlikely to address including spurious correlations (both in pre-training and fine-tuning datasets) and extrapolation across time.

B Conclusion

In this work, we study the failure modes that pre-training *can* and *cannot* address. Our findings suggest that pre-training can help mitigate failures caused by poor extrapolation (e.g., inability to generalize to a new domain) but might not address other failures, such as those stemming from dataset biases. In light of this observation, dataset biases present a fundamental limitation that cannot be overcome by additional pre-training data or larger models. We thus encourage practitioners not to treat pre-training as a panacea for robustness. Instead, they should consider the specific failures modes they might encounter (i.e., ask their distribution shift) to determine if pre-training can help.

C Additional Discussion

C.1 Alternative fine-tuning strategies

In this work, we focus on the common setting in which a pre-trained model is fully fine-tuned. It is important to note that pre-trained models used in a zero-shot context (i.e., without fine-tuning) and partially fine-tuned models (e.g., only the final classification layer is updated) are frequently more robust than fully fine-tuned models [11, 10, 46]. Such models may have higher effective robustness than fully fine-tuned models or in some cases may even outperform fully fine-tuned models on the shifted distribution. However, such models are typically less performant on the reference distribution than fully fine-tuned models.

Several works observe this tradeoff between performance on the reference distribution and robustness and devise methods for mitigating it, i.e., methods for *robust fine-tuning* [12, 47, 46]. For example, Kumar et al. [46] argue that full fine-tuning "distorts" pre-trained features and propose linear probing *before* full fine-tuning (LP-FT) to prevent distortion. They also suggest that fine-tuning a model initialized as a zero-shot classifier may have a similar effect. In addition to full fine-tuning, in Section 4 we thus consider LP-FT and zero-shot initialization for fine-tuning. On in-support shifts, we observe that LP-FT and zero-shot initialization do not provide effective robustness benefits compared to full fine-tuning (see Figure 2), suggesting that these strategies do not help mitigate dataset biases.

Another strategy for robust fine-tuning is to ensemble a zero-shot model and a fully fine-tuned model. Both weight-space ensembles [12] and output-space ensembles [47] have been shown to improve robustness, sometimes even without sacrificing performance on the reference distribution. In fact, this strategy can yield robustness benefits even when dataset biases are a primary failure mode because the zero-shot model is independent of the biased reference dataset. Our work seeks to complement such empirically effective strategies by providing an understanding of when they are necessary. In particular, our findings suggest that ensembling is valuable precisely when dataset biases cause failures.

C.2 Can pre-training hurt extrapolation?

In this work, we discuss distribution shifts in which pre-training is beneficial to a model's ability to extrapolation outside of the reference distribution. A natural question to consider is whether pre-training can instead *hurt* it, yielding worse extrapolation than a model trained from scratch. A recent work by Salman et al. [29] suggests that this is indeed possible. Specifically, they show that biases of pre-trained models can persist during fine-tuning. For example, a model pre-trained on ImageNet and fine-tuned on CIFAR-10 is highly sensitive to the presence of tennis balls (which are an ImageNet class but not a CIFAR-10 class). Meanwhile, a model trained from scratch on CIFAR-10 is not particularly sensitive to tennis balls. Thus, under a hypothetical "tennis ball shift" in which tennis balls appear in images in the shifted distribution, a pre-trained model would be less robust than a model trained from scratch. In this instance, pre-training provides a *harmful* prior for how to extrapolate.

C.3 Understanding the robustness of pre-trained language models to spurious correlations

Tu et al. [43] study the robustness of pre-trained language models to distribution shifts with spurious correlations. Their central finding is that pre-training *can* improve performance on shifted datasets in which spurious correlations do not hold. They illustrate that this is because pre-trained models can generalize better from the small number of counterexamples to these correlations in the reference dataset. This is a similar phenomenon to our observation from Section 4 that pre-training can provide limited effective robustness even on in-support shifts (which we further explain in Section E.1.2). In cases such as those discussed by Tu et al. [43], we hypothesize that pre-training can help to a limited extent by extrapolating better, but cannot mitigating the underlying failure mode of dataset biases.

D Theoretical Results

D.1 Proof of Theorem 3.1

Setup. Suppose that we are given access to a reference dataset S_{ref} of input-label pairs (x, y), with $x \in \mathbb{R}^d$ and $y \in \{-1, 1\}$. We decide to learn a linear classifier for this task by finding a weight w that minimizes the (standard) logistic loss on S_{ref}:

$$L_{\text{ref}}(w) = \sum_{(x,y)\in S_{\text{ref}}} \log(1 + e^{-w^\top x \cdot y}). \qquad \textbf{(??)}$$

We assume that the reference dataset S_{ref} satisfies the following conditions:

1. **Inputs in S_{ref} lie within a k-dimensional (with $k < d$) subspace W_{ref} of $\mathbb{R}^d$.** Intuitively, this condition represents a lack of variation in certain features in the reference dataset (e.g., if the reference dataset has fixed lighting conditions).
2. **The logistic loss L_{ref} has a minimum value.** This condition ensures that minimizing L_{ref} is well-defined, and means (roughly) that the two classes are not linearly separable. Note that there may be multiple weights that attain this minimum value.

Theorem 3.1. *Suppose that we start with initial weights $w_{init} \in \mathbb{R}^d$ and run gradient descent to minimize the logistic loss $L_{ref}(w)$ on S_{ref}. With an appropriately chosen learning rate, gradient descent converges to weights $\hat{w}$ that minimize L_{ref}. Furthermore, $\hat{w}$ can be written as*

$$\hat{w} = w^*_{ref} + (w_{init} - \mathit{proj}_{W_{ref}} w_{init}) \qquad (1)$$

where w^*_{ref} does not depend on the initialization w_{init} and lies within the reference subspace W_{ref}.

To prove Theorem 3.1, we will first show that running gradient descent starting from an initialization within W_{ref} always converges to the same weights w^*_{ref}. We will then show that running gradient descent starting from an arbitrary initialization has the same convergence behavior except for an "offset" term $(w_{\text{init}} - \text{proj}_{W_{\text{ref}}} w_{\text{init}})$ representing the component of the initialization that lies outside of W_{ref}.

D.1.1 Convexity and smoothness of the loss

We begin by providing the gradient and hessian of L_{ref} and using these to establish convexity (Lemma D.1) and smoothness (Lemma D.2) properties of L_{ref}. The gradient of L_{ref} is

$$\nabla L_{\text{ref}}(w) = \sum_{(x,y)\in S_{\text{ref}}} x \cdot y \cdot \frac{1}{1 + e^{w^\top x \cdot y}}. \qquad (2)$$

The Hessian of L_{ref} is

$$\nabla^2 L_{\text{ref}}(w) = \sum_{(x,y)\in S_{\text{ref}}} x x^\top \cdot \frac{1}{2 + e^{-w^\top x \cdot y} + e^{w^\top x \cdot y}} = X^\top D(w) X \qquad (3)$$

where $X \in \mathbb{R}^{|S_{\text{ref}}| \times d}$ is the matrix of inputs in S_{ref} and $D(w) \in \mathbb{R}^{|S_{\text{ref}}| \times |S_{\text{ref}}|}$ is the diagonal matrix with $D(w)_{ii} = \frac{1}{2 + e^{-w^\top x \cdot y} + e^{w^\top x \cdot y}}$. Note in particular that the non-zero elements of $D(w)$ are in $(0, 1/4)$.

Lemma D.1. *The loss L_{ref} is (1) convex on $\mathbb{R}^d$, (2) strictly convex on W_{ref}, and (3) strongly convex on any closed convex subset of W_{ref}.*

Proof. According to Taylor's Theorem, for any $u, v \in \mathbb{R}^d$, there exists a $\alpha \in [0, 1]$ such that

$$L_{\text{ref}}(v) = L_{\text{ref}}(u) + \nabla L_{\text{ref}}(u)^\top (v - u) + \frac{1}{2} \cdot (v - u)^\top \nabla^2 L_{\text{ref}}(v + \alpha \cdot (v - u))(v - u). \qquad (4)$$

1. **Convexity on $\mathbb{R}^d$.** To show that L_{ref} is convex on $\mathbb{R}^d$, we need to show that

$$L_{\text{ref}}(v) \geq L_{\text{ref}}(u) + \nabla L_{\text{ref}}(u)^\top (v - u)$$

for any $u, v \in \mathbb{R}^d$. Using (4), it suffices to show that $a^\top \nabla^2 L_{\text{ref}}(w) a \geq 0$ for any $a \in \mathbb{R}^d$ and $w \in \mathbb{R}^d$. Recall from (3) that $\nabla^2 L_{\text{ref}}(w) = X^\top D(w) X$. Thus, we have

$$\begin{aligned} a^\top \nabla^2 L_{\text{ref}}(w) a &= a^\top X^\top D(w) X a \\ &= \|D(w)^{1/2} X a\|_2^2 \\ &\geq 0 \end{aligned}$$

2. **Strict convexity on W_{ref}.** Next, to show that L_{ref} is strictly convex on W_{ref}, we need to show that

$$L_{\text{ref}}(v) > L_{\text{ref}}(u) + \nabla L_{\text{ref}}(u)^\top (v - u)$$

for any $u, v \in W_{\text{ref}}$. Using (4), it suffices to show that $a^\top \nabla^2 L_{\text{ref}}(w) a > 0$ for any non-zero $a \in W_{\text{ref}}$ and $w \in W_{\text{ref}}$. We know that $a^\top \nabla^2 L_{\text{ref}}(w) a = \|D(w)^{1/2} X a\|_2^2$. Since $D(w)$ is diagonal with positive entries along the diagonal, $\|D(w)^{1/2} X a\|_2^2 > 0$ if and only if $Xa \neq 0$. Recall that W_{ref} is the subspace spanning the rows of X. Hence, since a is non-zero and is in W_{ref}, we know that $Xa \neq 0$.

3. **Strong convexity on any closed convex subset of W_{ref}.** Finally, to show that L_{ref} is strongly convex on any closed convex subset T of W_{ref}, we need to show that there exists an $m > 0$ such that

$$L_{\text{ref}}(v) \geq L_{\text{ref}}(u) + \nabla L_{\text{ref}}(u)^\top (v - u) + \frac{m}{2} \|v - u\|_2^2$$

for any $u, v \in T$. Using (4), it suffices to show that there exists an $m > 0$ such that $a^\top \nabla^2 L_{\text{ref}}(w) a > \frac{m}{2} \cdot \|a\|_2^2$ for any $a \in W_{\text{ref}}$ and $w \in T$. Making use of the fact that T is closed, let λ_{min} be the minimum diagonal entry of $D(w)$ for $w \in T$, that is,

$$\lambda_{\text{min}} = \min_{w \in T} \min_{i \in \{1, \ldots, |S_{\text{ref}}|\}} D(w)_{ii}.$$

Next, let c_{min} be the minimum value of $\|Xa\|_2^2$ over unit vectors a in W_{ref}, that is,

$$c_{\text{min}} = \min_{a \in W_{\text{ref}}, \|a\|_2 = 1} \|Xa\|_2^2.$$

We previously established that $Xa \neq 0$ for any non-zero $a \in W_{\text{ref}}$, which means that $c_{\text{min}} > 0$. Finally, we conclude that for $m = 2 \cdot \lambda_{\text{min}} \cdot c_{\text{min}}$, $a^\top \nabla^2 L_{\text{ref}}(w) a = \|D(w)^{1/2} X a\|_2^2 \geq \lambda_{\text{min}} \cdot c_{\text{min}} \cdot \|a\|_2^2 = \frac{m}{2} \cdot \|a\|_2^2$.

□

Lemma D.2. *The gradient of the loss function ∇L_{ref} is K-Lipschitz with $K = \|X\|_{op}^2 / 4$.*

Proof. To show that ∇L_{ref} is K-Lipschitz, we need to show that $\nabla^2 L_{\text{ref}}(w) \preceq KI$. Recall from (3) that $\nabla^2 L_{\text{ref}}(w) = X^\top D(w) X$. Thus, we have

$$\begin{aligned} a^\top \nabla^2 L_{\text{ref}}(w) a &= a^\top X^\top D(w) X a \\ &= \|D(w)^{1/2} X a\|_2^2 \\ &\leq \|D(w)^{1/2}\|_{\text{op}}^2 \cdot \|X\|_{\text{op}}^2 \cdot \|a\|_2^2 \\ &\leq (\|X\|_{\text{op}}^2 / 4) \cdot \|a\|_2^2. \end{aligned}$$

In the final step, we use the fact that $D(w)$ is diagonal with non-zero elements in $(0, 1/4)$ to conclude that $\|D(w)^{1/2}\|_{\text{op}}^2 \leq 1/4$. □

D.1.2 Convergence of gradient descent within the reference subspace

Next, we establish that there exists a unique minimumizer of L_{ref} within the reference subspace W_{ref} (Lemma D.3) and that gradient descent converges to these weights (Lemma D.4).

Lemma D.3. *There exists a unique $w_{ref}^* \in W_{ref}$ such that $w_{ref}^* \in \arg\min_w L(w)$.*

Proof. We will first show that there exists a $w^*_{\text{ref}} \in W_{\text{ref}}$ such that $w^*_{\text{ref}} \in \arg\min_w L_{\text{ref}}(w)$. Let $w^* \in \arg\min_w L_{\text{ref}}(w)$ be an arbitrary minimimum point of L_{ref}. By definition, for every $(x, y) \in S_{\text{ref}}$, $x \in W_{\text{ref}}$. Hence, for every such x, $w^\top x = \text{proj}_{W_{\text{ref}}} w^\top x$. This means that $L_{\text{ref}}(w^*) = L_{\text{ref}}(\text{proj}_{W_{\text{ref}}} w^*)$, which implies that $w^*_{\text{ref}} := \text{proj}_{W_{\text{ref}}} w^* \in \arg\min_w L_{\text{ref}}(w)$, as desired. Next, because L_{ref} is strictly convex on W_{ref} (Lemma D.1), w^*_{ref} is the only minimum point of L_{ref} in W_{ref}. □

Lemma D.4. *If we start with $w_{init} \in W_{ref}$ and run gradient descent with $\eta = 4/\|X\|^2_{op}$ to minimize $L_{ref}(w)$, the weights will converge to w^*_{ref}.*

Proof. Suppose that we start with initial weights $w_{\text{init}} \in W_{\text{ref}}$ and run gradient descent to minimize L_{ref} with learning rate η. In particular, let $w^{(0)} = w_{\text{init}}$ and $w^{(t+1)} = w^{(t)} + \eta \cdot \nabla L_{\text{ref}}(w^{(t)})$. Because L_{ref} is convex (Lemma D.1), ∇L_{ref} is K-Lipschitz with $K = \|X\|^2_{\text{op}}/4$ (Lemma D.2), and $\eta = 4/\|X\|^2_{\text{op}} \leq 1/K$, we know from Theorem 3.2 of [48] that

$$L_{\text{ref}}(w^{(t)}) - L_{\text{ref}}(w^*_{\text{ref}}) \leq \frac{K \cdot \|w_{\text{init}} - w^*_{\text{ref}}\|}{t-1}. \tag{5}$$

Hence, the loss attained by $w^{(t)}$ converges to the optimal loss attained by w^*_{ref}. To show that $w^{(t)}$ converges to w^*_{ref}, we will show that L_{ref} is strongly convex on a set containing every $w^{(t)}$ for $t \geq 0$. In particular, consider the set $W_{\text{GD}} = \{w \in W_{\text{ref}} \mid \|w - w^*_{\text{ref}}\|_2 \leq \|w_{\text{init}} - w^*_{\text{ref}}\|_2\}$ containing weights in W_{ref} at least as close to w^*_{ref} as w_{init}. Clearly, W_{GD} contains $w^{(0)} = w_{\text{init}}$. We know from Theorem 3.2 of [48] that with each iteration of gradient descent we get closer to a minimum point, that is, $\|w^{(t+1)} - w^*_{\text{ref}}\| \leq \|w^{(t)} - w^*_{\text{ref}}\|$. Additionally, because w_{init} and ∇L_{ref} are in W_{ref}, every $w^{(t)}$ is in W_{ref}. Hence, every $w^{(t)}$ is in W_{GD}. Because W_{GD} is closed and convex, from Lemma D.1 we know that L_{ref} is strongly convex on W_{GD}. This means that there exists an $m > 0$ such that

$$L_{\text{ref}}(w^{(t)}) \geq L_{\text{ref}}(w^*_{\text{ref}}) + \nabla L_{\text{ref}}(w^*_{\text{ref}})^\top (w^{(t)} - w^*_{\text{ref}}) + \frac{m}{2} \cdot \|w^{(t)} - w^*_{\text{ref}}\|^2_2.$$

Plugging in $\nabla L_{\text{ref}}(w^*_{\text{ref}}) = 0$ and rearranging, we get

$$\|w^{(t)} - w^*_{\text{ref}}\|^2_2 \leq \frac{2}{m} \cdot (L_{\text{ref}}(w^{(t)}) - L_{\text{ref}}(w^*_{\text{ref}})).$$

Finally, combining with (5) yields

$$\|w^{(t)} - w^*_{\text{ref}}\|^2_2 \leq \frac{2 \cdot K \cdot \|w_{\text{init}} - w^*_{\text{ref}}\|}{m \cdot (t-1)} \tag{6}$$

which completes our proof. □

D.1.3 Proof of Theorem 3.1

We are now ready to prove Theorem 3.1. Suppose that we start with initial weights w_{init} and run gradient descent to minimize L_{ref} with learning rate $\eta = 4/\|X\|^2_{\text{op}}$. In particular, let $w^{(0)} = w_{\text{init}}$ and $w^{(t+1)} = w^{(t)} + \eta \cdot \nabla L_{\text{ref}}(w^{(t)})$ for $t \geq 0$. We will show that running gradient descent starting with an arbitrary w_{init} has the same behavior as running gradient descent with w_{init} projected onto W_{ref}. To be more precise, suppose that we instead start with initial weights $\text{proj}_{W_{\text{ref}}} w_{\text{init}}$ when running gradient descent. In particular, let ${w_{\text{proj}}}^{(0)} = \text{proj}_{W_{\text{ref}}} w_{\text{init}}$ and ${w_{\text{proj}}}^{(t+1)} = {w_{\text{proj}}}^{(t)} + \eta \cdot \nabla L_{\text{ref}}({w_{\text{proj}}}^{(t)})$ for $t \geq 0$. Then the trajectory of $w^{(t)}$ is the same as that of ${w_{\text{proj}}}^{(t)}$ but with an additional component $(w_{\text{init}} - \text{proj}_{W_{\text{ref}}} w_{\text{init}})$, that is,

$$w^{(t)} = (w_{\text{init}} - \text{proj}_{W_{\text{ref}}} w_{\text{init}}) + {w_{\text{proj}}}^{(t)}.$$

To show that this is the case, we will proceed by induction. As a base case,

$$\begin{aligned} w^{(0)} &= w_{\text{init}} \\ &= w_{\text{init}} - \text{proj}_{W_{\text{ref}}} w_{\text{init}} + \text{proj}_{W_{\text{ref}}} w_{\text{init}} \\ &= (w_{\text{init}} - \text{proj}_{W_{\text{ref}}} w_{\text{init}}) + {w_{\text{proj}}}^{(0)}. \end{aligned}$$

For the inductive step, assume that the statement holds for $t = k$. Then,

$$\begin{aligned} w^{(k+1)} &= w^{(k)} + \eta \cdot \nabla L_{\text{ref}}(w^{(k)}) \\ &= (w_{\text{init}} - \text{proj}_{W_{\text{ref}}} w_{\text{init}}) + {w_{\text{proj}}}^{(k)} + \eta \cdot \nabla L_{\text{ref}}((w_{\text{init}} - \text{proj}_{W_{\text{ref}}} w_{\text{init}}) + {w_{\text{proj}}}^{(k)}) \\ &= (w_{\text{init}} - \text{proj}_{W_{\text{ref}}} w_{\text{init}}) + {w_{\text{proj}}}^{(k)} + \eta \cdot \nabla L_{\text{ref}}({w_{\text{proj}}}^{(k)}) \\ &= (w_{\text{init}} - \text{proj}_{W_{\text{ref}}} w_{\text{init}}) + {w_{\text{proj}}}^{(k+1)} \end{aligned}$$

where in the third step we use the fact that $\nabla L_{\text{ref}}(w) = \nabla L_{\text{ref}}(\text{proj}_{W_{\text{ref}}} w)$. This completes the induction. Because ${w_{\text{proj}}}^{(0)} = \text{proj}_{W_{\text{ref}}} w_{\text{init}} \in W_{\text{ref}}$, from Lemma D.4 (in particular, from (6)), we know that

$$\|{w_{\text{proj}}}^{(t)} - w^*_{\text{ref}}\|_2^2 \leq \frac{2 \cdot K \cdot \|\text{proj}_{W_{\text{ref}}} w_{\text{init}} - w^*_{\text{ref}}\|}{m \cdot (t-1)}.$$

where K and m are positive constants. Finally, we conclude that

$$\begin{aligned} \|w^{(t)} - \hat{w}\|_2^2 &= \|((w_{\text{init}} - \text{proj}_{W_{\text{ref}}} w_{\text{init}}) + {w_{\text{proj}}}^{(t)}) - ((w_{\text{init}} - \text{proj}_{W_{\text{ref}}} w_{\text{init}}) + w^*_{\text{ref}})\|_2^2 \\ &= \|{w_{\text{proj}}}^{(t)} - w^*_{\text{ref}}\|_2^2 \\ &\leq \frac{2 \cdot K \cdot \|\text{proj}_{W_{\text{ref}}} w_{\text{init}} - w^*_{\text{ref}}\|}{m \cdot (t-1)} \end{aligned}$$

Hence, $w^{(t)}$ converges to $\hat{w}$, completing our proof.

E Additional Results

E.1 Evaluating the empirical robustness benefits of pre-training

E.1.1 How does the choice of fine-tuning hyperparameters affect robustness?

In Section 4, we select hyperparameters (in particular, learning rate) for fine-tuning that maximize accuracy on the reference distribution. This reasonably simulates hyperparameter selection in practice because typically only samples from the reference distribution are available.

In this section, we investigate how the choice of hyperparameters affects the robustness of pre-trained models. In particular, we would like to understand if pre-training yields little effective robustness to in-support shifts and substantial effective robustness to out-of-support shifts across a wider range of hyperparameter choices. We study the tint shift (an in-support shift) and the pad shift (an out-of-support shift) from Section 4 and vary the learning rate, weight decay, number of epochs, and batch size of a CLIP ViT-B/32 initialized with zero-shot weights (Figure 4). With zero-shot initialization, the starting point of fine-tuning is a robust model that performs well on our task. Hence, even under an in-support shift, hyperparameter choices that do not change the model substantially (e.g., low learning rate, small number of epochs) result in substantial effective robustness. However, these hyperparameter choices generally result in lower absolute reference and shifted accuracies, and are thus unreasonable. The hyperparameter choices that are relevant in practice are those with high reference accuracy, and these are the hyperparameters that we use in our experiments.

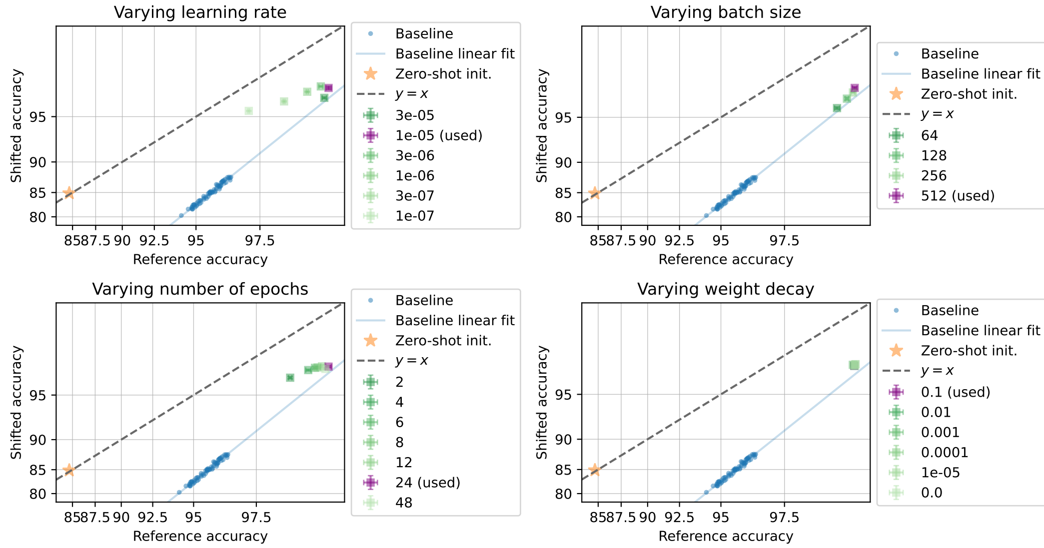
E.1.2 Why do pre-trained models exhibit any effective robustness under in-support shifts?

On the left side of Figure 2, we observe a small robustness gap between pre-trained models and models trained from scratch on in-support shifts constructed with CIFAR-10. This seemingly refutes our intuition that pre-training helps specifically with extrapolation, which implies that in-support, pre-training should offer *no* robustness benefits. However, we hypothesize that this robustness gap actually stems from extrapolation: under empirical shifts (where we have access to samples and not an actual distribution), models must extrapolate from the limited number of samples available. For example, in the label shift setting there are only $1,250$ instances of each of the minority classes in the training dataset, and these samples may not fully represent the distributions of these classes.

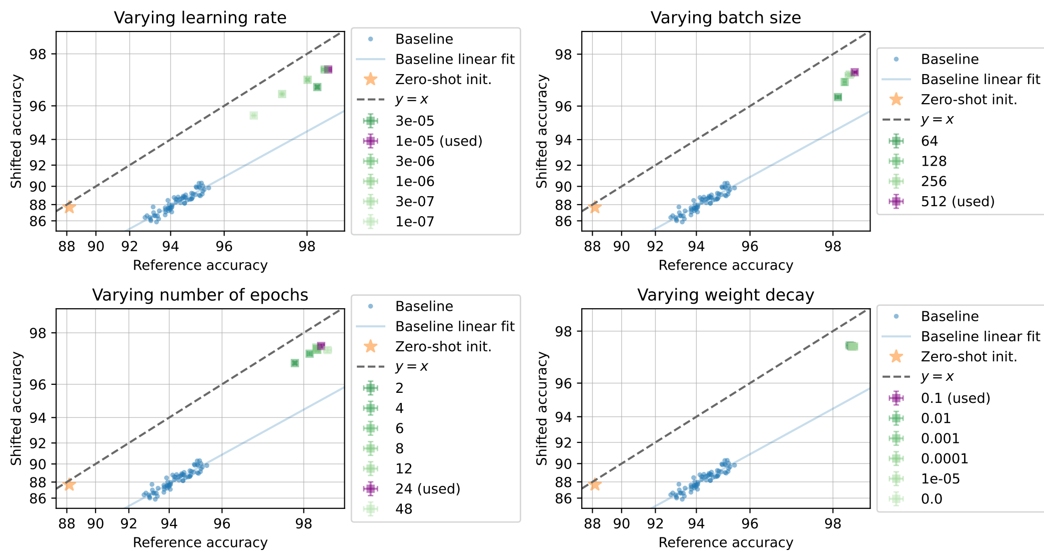
To determine whether the limited sample size explains the robustness gap, we repeat our experiment from Section 4 on 1 million examples (a $20\times$ increase in size) from the CIFAR-5m dataset [49], a large synthetically generated dataset resembling CIFAR-10. Besides the number of epochs, which we decrease due to the larger dataset size, we use the same hyperparameters as in the original experiment and again select the learning rate for fine-tuning that maximizes reference accuracy. We observe that with a larger sample size, the effective robustness of pre-trained models on in-support shifts vanish (Figure 5). This suggests that under the CIFAR-10 in-support shifts, the small effective robustnesses of pre-trained models are due to extrapolation from the small number of samples available, which would be consistent with our hypothesis. We also observe decreases in the effective robustnesses of pre-trained models on out-of-support shifts, though pre-trained models still exhibit substantial robustness under these shifts. This may be because as the size of the fine-tuning dataset increases, the pre-trained features change more during fine-tuning, leading to lower robustness.

E.1.3 Dividing natural shifts into in-support and out-of-support splits

In Section 4, we constructed synthetic in-support and out-of-support shifts and observed that pre-training can significantly improve robustness to the latter but not the former. Now, we demonstrate that this principle seems to extend to natural shifts as well. Note that it is hard to find natural shifts that are "purely" in-support. After all, under natural shifts the shifted dataset may contain some inputs that are similar to those in the reference dataset and some that are not. For example, in a shift from photos to sketches, some sketches may look more photorealistic but most would probably be clearly distinguishable from photos. To measure robustness to each type of shift, we thus *divide* several natural shifted datasets each into an "in-support split" containing inputs that look like they could have come from the reference dataset and an "out-of-support split" containing the remaining inputs. We do so by training a classifier to distinguish between the reference and shifted datasets and using this classifier to approximate the probability of sampling a given shifted example from the reference distribution (see Appendix F.3.1 for the details of the splitting method).



(a) **In-support shift.** The in-support shift we consider is the “tint shift” in which we introduce a tint that is spuriously correlated with the label. On this in-support shift, learning rate and number of epochs influence effective robustness, but the best hyperparameter choices result in a model with little effective robustness.



(b) **Out-of-support shift.** The out-of-support shift we consider is the “pad shift” in which we pad images in the shifted distribution. On this out-of-support shift, batch size most significantly affects robustness, while learning rate and number of epochs affect overall performance.

Figure 4: **The effects of hyperparameter choices on robustness.** We vary hyperparameters when fine-tuning a CLIP ViT-B/32 initialized with zero-shot weights on synthetic CIFAR-10 shifts from Section 4 (different shades of green). Varying certain hyperparameters (e.g., learning rate, number of epochs) can affect the effective robustness of pre-trained models even on an in-support shift. In our experiments, we choose hyperparameters which yield high reference accuracy (purple).

Specifically, in our study, we consider three natural shifts of the ImageNet dataset: ImageNet-V2 [50], which closely resembles ImageNet, ImageNet Sketch [51], which consists of sketches of ImageNet classes, and ImageNet-R [30], which consists of “renditions” (e.g, paintings, sculptures, cartoons) of a subset of ImageNet classes. We choose these shifted datasets because they include many inputs that look like they could have come from ImageNet and many that do not (according to our splitting

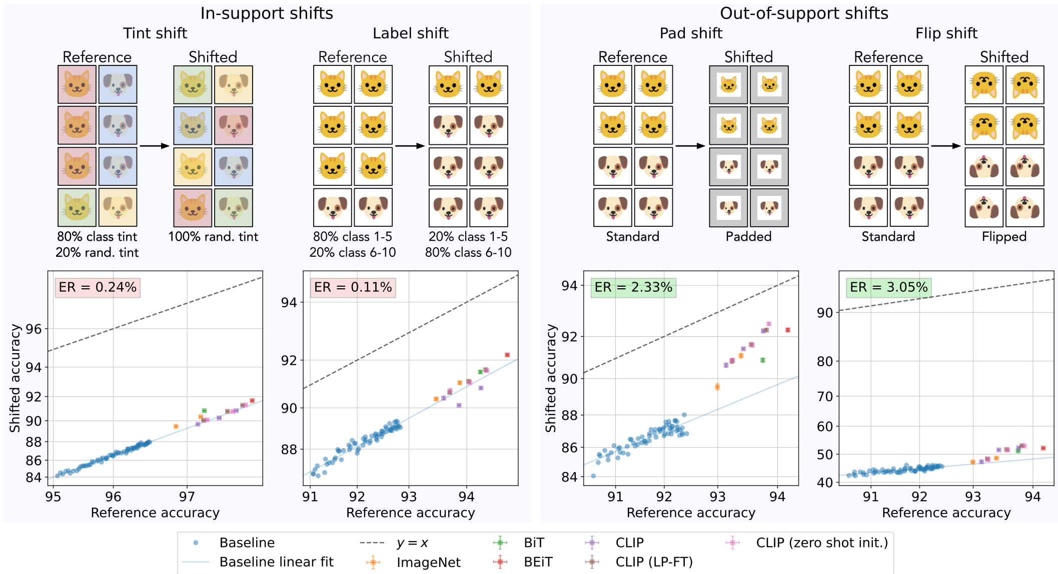
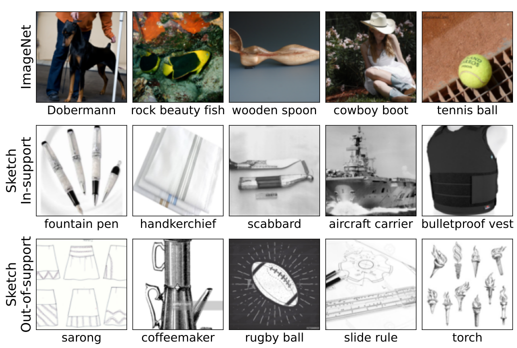
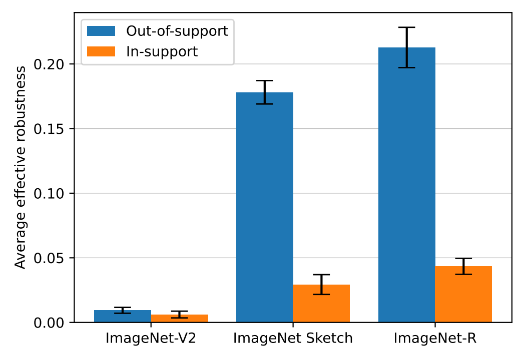


Figure 5: **Robustness of pre-trained models to synthetic in-support and out-of-support shifts constructed from CIFAR-5m.** Pre-trained models exhibit virtually no effective robustness on the in-support shifts (left), but have significant effective robustness on the out-of-support shifts (right). Error bars denote 95% confidence intervals over 8 random trials.



(a) Random samples from ImageNet (top) as well as the in-support split (middle) and out-of-support split (bottom) of ImageNet Sketch.



(b) Average effective robustness of 24 pre-trained models on each split of each of the three shifts. Error bars denote 95% confidence intervals.

Figure 6: We divide each of the ImageNet-V2, ImageNet Sketch and ImageNet-R datasets into an in-support split containing examples that look like ImageNet examples and an out-of-support split containing examples that look unlike ImageNet examples. We display samples from each split of ImageNet Sketch in Figure 6a and report the average effective robustnesses of pre-trained models in Figure 6b. See Figure 8 for scatterplots of reference vs. shifted accuracy.

method)[2]. In Figure 6a, we visualize examples from the in-support and out-of-support splits of ImageNet Sketch.

Consistently with our hypothesis that pre-training helps specifically with extrapolation, on the out-of-support splits of ImageNet Sketch and ImageNet-R pre-trained models have substantially higher effective robustness than on the respective in-support splits (see Figure 6b). On both ImageNet-V2 splits, however, pre-trained models have very little effective robustness. This may be because ImageNet-V2 is visually similar to ImageNet, so poor extrapolation might not be a significant failure

[2]We also explored ObjectNet [52] and ImageNet-Vid-Robust [53] but our splitting method marks fewer than 50 examples from these shifted datasets as "in-support," and thus we cannot reliably measure in-support accuracy.

Table 1: Sizes of in-support and out-of-support splits.

Dataset	In-support split size	Out-of-support split size
ImageNet-V2	1920	8080
ImageNet Sketch	162	50727
ImageNet-R	588	29412

mode (instead, the performance drop may be due to an increased presence of "harder" examples, as Recht et al. [50] suggest). Thus, if pre-training helps only with extrapolation, it would not be able to substantially improve robustness on the ImageNet-V2 out-of-support examples. See Appendix F.3.2 for a description of the exact setup.

Sizes of in-support and out-of-support splits. In Table 1, we report the sizes of the in-support and out-of-support splits we compute for ImageNet-V2, ImageNet Sketch and ImageNet-R. The out-of-support splits are much larger than the in-support splits, perhaps because the large majority of the examples from these shifted datasets look unlike examples from ImageNet.

Additional examples from in-support and out-of-support splits. In Figure 7, we provide samples from the in-support and out-of-support splits we compute for ImageNet-V2, ImageNet-Sketch and ImageNet-R.

Scatter plots of reference vs. shifted accuracy. In Figure 8, we provide scatter plots of accuracy on ImageNet vs. accuracy on the in-support and out-of-support splits of ImageNet-V2, ImageNet Sketch and ImageNet-R.

Controlling for difficulty when measuring effective robustness. The significance of a given effective robustness depends on the "difficulty" of a distribution shift. For example, if a shift causes an accuracy drop of 5%, an effective robustness of 4% might be considered large, but if a shift that causes a drop of 25%, an effective robustness of 4% would probably be considered small. When we divide a shifted dataset into an in-support and out-of-support split, the out-of-support split is typically more difficult than the in-support split. If we compare the effective robustness of pre-trained models on examples of similar difficulty in the in-support and out-of-support splits, do our findings still hold? In particular, do pre-trained models still exhibit substantially higher robustness on out-of-support examples than on in-support examples?

To answer this question, we re-weight examples in out-of-support splits such that the difficulty distribution of the out-of-support split matches that of the in-support split. Specifically, we quantify the difficulty of a given example in terms of the fraction of baseline models (of 77 total baseline models) that classify it incorrectly. Given an example of difficulty d, we re-weight it by a factor of $p_{\text{in-support}}(d)/p_{\text{out-of-support}}(d)$ where $p_{\text{in-support}}$ is the difficulty probability density function of the in-support split and $p_{\text{out-of-support}}$ is the difficulty probability density function of the out-of-support split. We then compute a "re-weighted" accuracy, which in turn yields a re-weighted effective robustness, on the out-of-support split. Intuitively, this re-weighted effective robustness represents the effective robustness of pre-trained models on out-of-support examples of similar difficulty to in-support examples.

We report the re-weighted effective robustnesses in Figure 9. We observe that the re-weighted effective robustnesses of pre-trained models on out-of-support splits are indeed lower than the original effective robustnesses. However, they are still substantially higher than the effective robustnesses on in-support splits.

E.2 Curating datasets for fine-tuning

E.2.1 Understanding the benefits of pre-training when fine-tuning on a curated dataset

In Section 5, we find that fine-tuning on a curated dataset with only 64 examples can yield a performant and robust model for hair color classification. We observe that pre-training is necessary for effective use of the small curated dataset; in particular, training a model from scratch on a curated

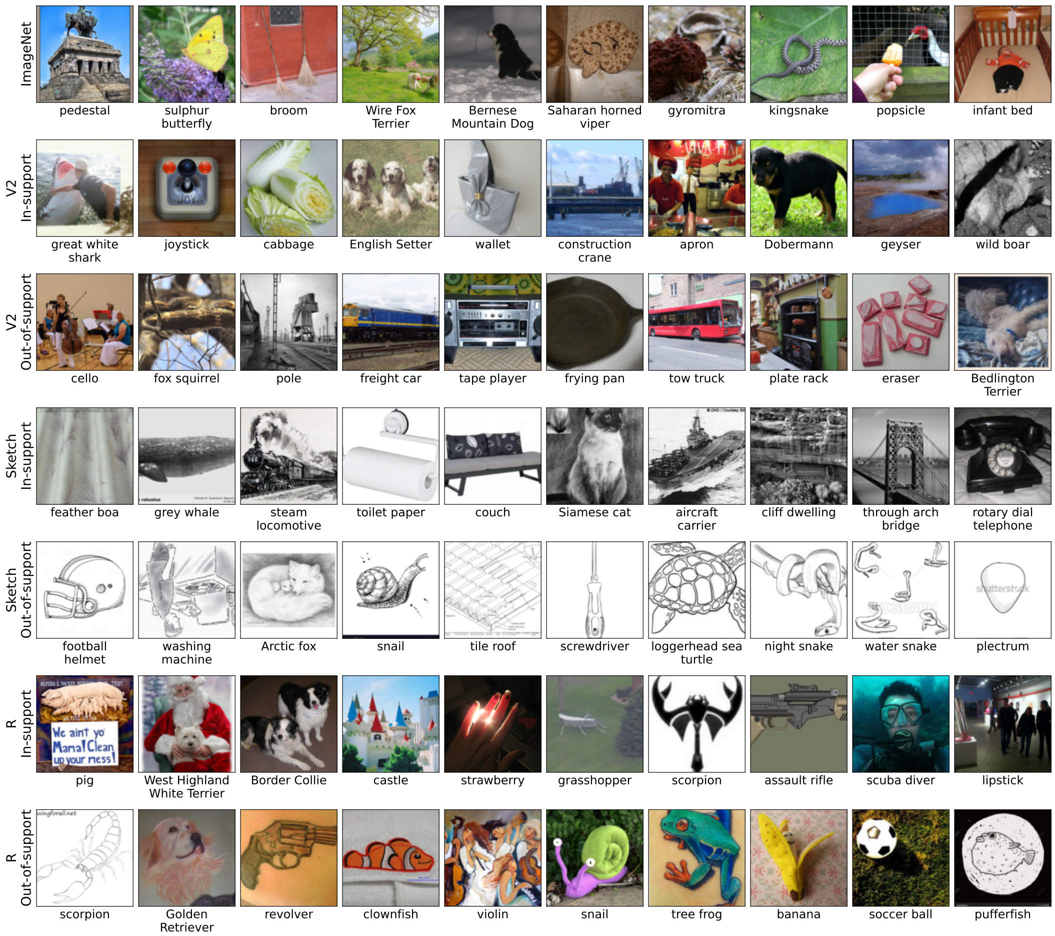


Figure 7: **Random samples from ImageNet and from the in-support and out-of-support splits of ImageNet-V2, ImageNet Sketch and ImageNet-R.** In ImageNet-V2, it is difficult to distinguish between examples from the in-support and out-of-support splits. In ImageNet Sketch and ImageNet-R, examples from the in-support splits look more realistic (i.e., more like ImageNet examples) than examples from the out-of-support splits.

dataset yields robustness gains, but these gains are smaller and many more examples are required to attain comparable accuracy.

In this section, we shed additional light on how pre-training helps in this setting. Based on our intuition from Sections 3 and 4 that pre-training helps specifically with extrapolation, we hypothesize that pre-training provides two benefits when training on a small curated dataset. First, a pre-trained model may be able to extrapolate better from a small number of examples. This would result in both higher accuracy on the original CelebA distribution and higher worst-group accuracy, which we observe in Figure 3b. Second, recall that our curated dataset consists entirely of females, but hair color classification models are expected to perform well on males too. To compare different model's ability to extrapolate along this axis, we plot the balanced accuracy on males against the balanced accuracy on females. In Figure 3a, we observe that the pre-trained model indeed generalizes better to males than models trained from scratch.

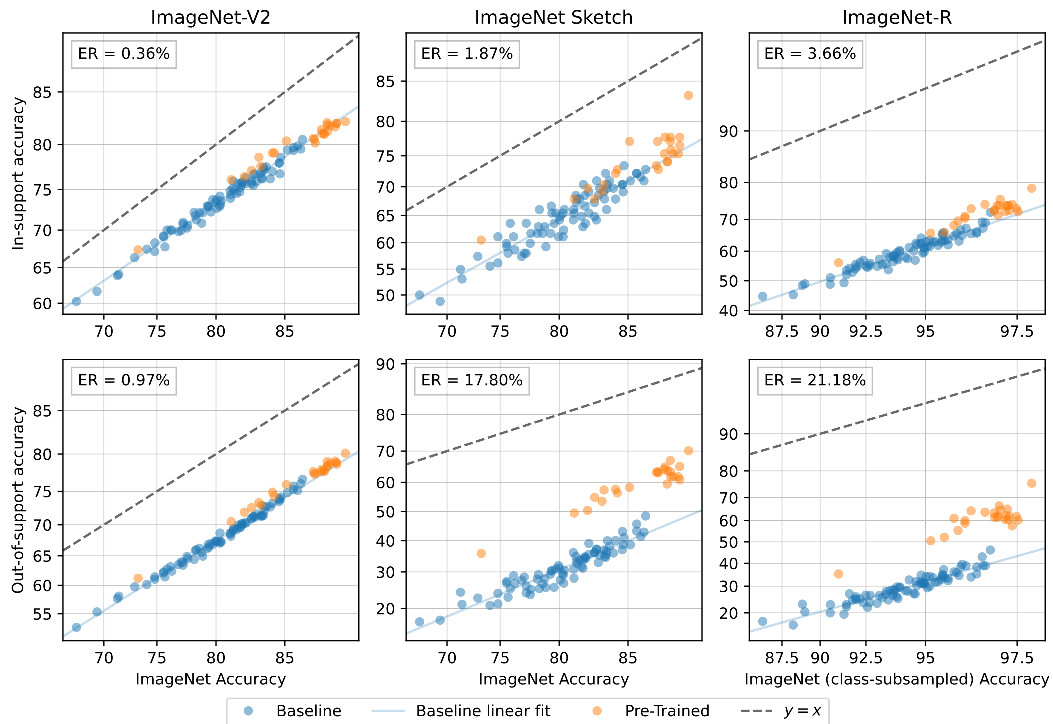


Figure 8: **Reference vs. shifted accuracy for in-support and out-of-support splits of ImageNet shifts.** On each of the three ImageNet shifts we consider, the average effective robustness (ER) of pre-trained models (orange) above the baseline of models trained from scratch (blue) on the in-support split (top) is small. Meanwhile, their effective robustness can be very large on the out-of-support split (bottom).

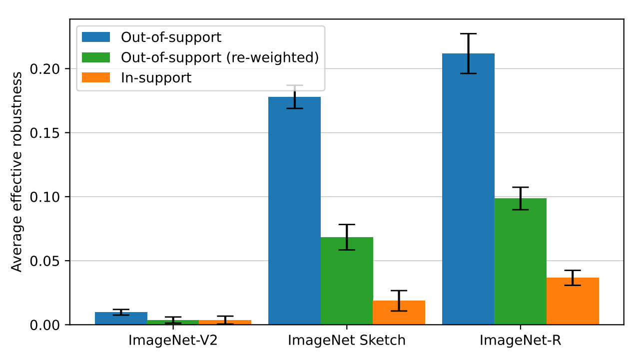


Figure 9: **Re-weighed effective robustness of pre-trained models on in-support and out-of-support splits of ImageNet shifts.** When we re-weight examples in out-of-support splits to match the difficulty distributions of their corresponding in-support splits, the average effective robustnesses of pre-trained models (green) decrease relative to the original effective robustnesses (blue). However, they are still very high on ImageNet Sketch and ImageNet-R. Meanwhile, the average effective robustnesses of pre-trained models on in-support splits (orange) are consistently low.

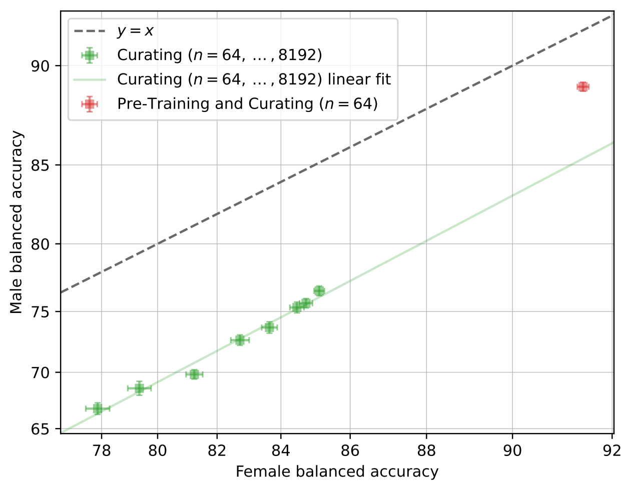


Figure 10: **Comparing extrapolation from females to males of pre-trained models and models trained from scratch.** We plot the balanced accuracy on males against the balanced accuracy of females of a pre-trained model fine-tuned on the curated dataset from Section 5 (red) and models trained from scratch on this dataset (green). Models trained from scratch establish a linear relationship between male and female balanced accuracy; however, the pre-trained model outperforms this trend, suggesting that it more effectively extrapolates to males from the female-only curated dataset.

F Experiment Details

F.1 General

F.1.1 Model training

All models are trained using the FFCV data-loading library [54] on a cluster of A100 GPUs.

F.1.2 Measuring effective robustness

Effective robustness. In this work, we quantify the robustness of pre-trained models using *effective robustness* (ER), a measure of the robustness a model above the "baseline" of models trained from scratch [8]. Computing this metric first involves establishing a relationship between the accuracies of baseline models (in our case, models trained from scratch on a reference dataset). In particular, let $\text{Acc}_{\text{ref}}(M)$ and $\text{Acc}_{\text{shift}}(M)$ denote the accuracies of a model M on test datasets drawn from the reference and shifted distributions, respectively. Given a set $\mathcal{M}_{\text{baseline}}$ of baseline models, we compute a linear fit relating $\Phi^{-1}(\text{Acc}_{\text{ref}}(M))$ and $\Phi^{-1}(\text{Acc}_{\text{shift}}(M))$, where Φ^{-1} is the probit function (i.e., the inverse cumulative distribution function of the standard normal distribution). We compute a linear fit relating probit-scaled accuracies because this has been empirically observed to improve the strength of the linear relationship [10, 8]. Formally, we compute parameters $\hat{a}$ and $\hat{b}$ such that

$$\hat{a}, \hat{b} = \arg\min_{a,b} \sum_{M \in \mathcal{M}_{\text{baseline}}} \|(a \cdot \Phi^{-1}(\text{Acc}_{\text{ref}}(M)) + b) - \Phi^{-1}(\text{Acc}_{\text{shift}}(M))\|_2.$$

Let $\widehat{\text{Acc}}_{\text{shift}}(M)$ be the resulting function estimating shifted accuracy given reference accuracy, that is

$$\widehat{\text{Acc}}_{\text{shift}}(M) = \Phi(\hat{a} \cdot \Phi^{-1}(\text{Acc}_{\text{ref}}(M)) + \hat{b}).$$

Then the effective robustness of a model M is

$$\text{ER}(M) = \widehat{\text{Acc}}_{\text{shift}}(M) - \text{Acc}_{\text{shift}}(M)$$

Intuitively, effective robustness is the extent to which a model's accuracy on the shifted distribution exceeds the accuracy of a baseline model with the same accuracy on the reference distribution.

Establishing a baseline for effective robustness. To establish a baseline with respect to which we can measure effective robustness, we train ResNet-18 models from scratch on differently sized subsets of the reference dataset. The number of models and the minimum subset size vary by experiment. Miller et al. [10] observe that models trained from scratch in this way often exhibit a strong linear relationship between their accuracies on the reference and shifted distributions (and the same relationship holds for models with different architectures, hyperparameters, etc.). In each of the experiments in which we measure effective robustness, we confirm that this relationship exists for our baseline models (see, e.g., Figure 2).

F.2 Exploring the empirical robustness benefits of pre-training

In Section 4, we measure the effective robustness of various pre-trained and fine-tuned models on two in-support and two out-of-support shifts synthetically constructed by modifying CIFAR-10.

Specifications of synthetic shifts. Here, we provide detailed descriptions of the four synthetic distribution shifts (see Figure 2 for visualizations).

1. **Tint shift** (in-support): We tint images (i.e., replace each pixel with a mix of the original value, with weight 0.75 and a specific color, with weight 0.25) such that the tint is correlated with the label in the reference distribution but not in the shifted distribution (i.e., tint is a spurious feature). Specifically, in the reference distribution we apply tint with a class-specific color to 80% of examples and a tint with a random color to the remaining 20%. Meanwhile, in the shifted distribution we apply a tint with a random color universally.
2. **Label shift** (in-support): Label shift is a commonly studied type of distribution shift in which the relative frequencies of classes change, but $p(x|y)$ is fixed. To construct a label shift, we sub-sample CIFAR-10 such that in the reference distribution, the first five classes are four times more likely to appear than the last five classes. In the shifted distribution, these relative frequencies are reversed.

3. **Pad shift** (out-of-support): We pad the images in the shifted distribution by adding 6 black pixels to each side of the original 32×32 CIFAR-10 images. Note that our models resize inputs to 224×224 as a pre-processing step, so padding does not affect the final size of images fed into our models.

4. **Flip shift** (out-of-support): We vertically flip images in the shifted distribution.

Shared model specifications. For data augmentation, we use the FFCV implementations of *RandomHorizontalFlip* and *RandomTranslate*. We preprocess images by resizing the original 32×32 images to a resolution of 224×224.

Specifications of baseline models. To establish a baseline, we train 64 models from scratch on subsets ranging from 50% of the dataset to the entire dataset. We train baseline models by running SGD for 96 epochs, using a triangular learning rate schedule with a peak learning rate of 0.5 and 5 warmup epochs, a batch size of 512, a weight decay of 5×10^{-4} and a momentum of 0.9.

Specifications of pre-trained models and fine-tuning strategies. We consider a 7 different pre-trained models (implementations from PyTorch Image Models [55]): ResNet-18 and ResNet-50 models trained on ImageNet-1K [1], a Big Transfer (BiT) [24] ResNet-50[3] model trained on ImageNet-21K, a BEiT [57] model with patch size of 16×16 and resolution of 224×224 trained on ImageNet-22K (with self-supervised training followed by standard supervised training), and CLIP models from Radford et al. [11] with ResNet-50, ViT-B/32 and ViT-B-16 architectures. For the ImageNet-1K models, BiT and BEiT models, we employ full fine-tuning with a randomly initialized classification layer. For the CLIP models, in addition to full fine-tuning we also consider linear probing followed by full fine-tuning (LP-FT) [46] and full fine-tuning initialized with zero-shot weights, as specified by Radford et al. [11]. This results in a total of 13 pre-trained and fine-tuned models, which we group in Figure 2 according to the pre-training and fine-tuning strategies (models within a single group have different architectures, but are otherwise the same).

We fine-tune models by running Adam for 24 epochs, using a cosine learning rate schedule with 5 warmup epochs. We select the best peak learning rate (in terms of reference accuracy) among $1 \times 10^{-2}, 3 \times 10^{-3}, 1 \times 10^{-3}, 3 \times 10^{-4}, 1 \times 10^{-4}, 3 \times 10^{-5}, 1 \times 10^{-5}, 3 \times 10^{-6}$. We use a batch size of 256 for the BiT, BEiT and ViT-B/16 models (due to memory constraints) and a batch size of 512 for other pre-trained models and use a weight decay of 0.1. When doing LP-FT, we perform linear probing with the same hyperparameters as full fine-tuning, except that we always use a learning rate of 0.1 and a weight decay of 0.

F.3 Dividing natural shifts into in-support and out-of-support splits

F.3.1 Splitting a Shifted Dataset

To split a shifted dataset into an "in-support split" and an "out-of-support split", we would ideally measure the reference distribution probability density p_{ref} of inputs in the shifted dataset and assign inputs with small p_{ref} to the out-of-support split. Unfortunately, it is difficult to estimate p_{ref} directly when dealing with high-dimensional inputs (in this case, images). Instead, we estimate the probability density *ratio* $p_{\text{ref}}/p_{\text{shift}}$, that is, how much more likely an input is under the reference distribution than under the shifted distribution. We then assign examples in the shifted dataset with $p_{\text{ref}}/p_{\text{shift}} < 0.2$ to the out-of-support split and examples with $p_{\text{ref}}/p_{\text{shift}} \geq 0.2$ to the in-support split. We visualize examples in Figure 7.

Estimating $p_{\text{ref}}/p_{\text{shift}}$. To estimate $p_{\text{ref}}/p_{\text{shift}}$, we use a classifier trained to distinguish between examples from the reference and shifted datasets. Specifically, let p be a probability mass/density function over examples that can either be drawn from $\mathcal{D}_{\text{ref}}$ or $\mathcal{D}_{\text{shift}}$ (i.e., p represents the distribution of a dataset created by joining a reference dataset and a shifted dataset). Next, let y_{ref} be the event that an example is drawn from $\mathcal{D}_{\text{ref}}$ and y_{shift} be the event that an example is drawn from $\mathcal{D}_{\text{shift}}$. We

[3] We use the ResNet-v2 variant of the ResNet-50 architecture from He et al. [56].

can express the ratio $p_{\text{ref}}/p_{\text{shift}}$ as follows:

$$\begin{aligned}\frac{p_{\text{ref}}(x)}{p_{\text{shift}}(x)} &= \frac{p(x|y_{\text{ref}})}{p(x|y_{\text{shift}})} \\ &= \frac{p(y_{\text{ref}}|x)\cdot p(x)}{p(y_{\text{ref}})}\cdot\frac{p(y_{\text{shift}})}{p(y_{\text{shift}}|x)\cdot p(x)} \\ &= \frac{p(y_{\text{ref}}|x)}{p(y_{\text{shift}}|x)}\cdot\frac{p(y_{\text{shift}})}{p(y_{\text{ref}})}.\end{aligned}$$

The terms $p(y_{\text{ref}})$ and $p(y_{\text{shift}})$ are easy to estimate since they are simply the proportions of reference and shifted examples in p. Hence, to estimate $p_{\text{ref}}/p_{\text{shift}}$ we just need to estimate $p(y_{\text{ref}}|x)$ and $p(y_{\text{shift}}|x)$.

To do so, we train a classifier to distinguish between reference and shifted examples on a dataset drawn from p. We construct such a dataset by combining $100K$ samples from ImageNet with each of the shifted datasets (for ImageNet-R, which contains a subset of the classes of ImageNet, we restrict the $100K$ samples to these classes). Next, we fine-tune a CLIP ViT-L/14 pre-trained on LAION-2B from OpenCLIP [58] to distinguish between reference and shifted examples. We first fine-tune just the final layer with a learning rate of 0.1 and then fine-tune the entire model with the best learning rate selected from $2\times10^{-4}, 1\times10^{-4}, 5\times10^{-5}, 2\times10^{-5}, 1\times10^{-5}, 5\times10^{-6}, 2\times10^{-6}$ and 1×10^{-6}. After training the classifier, we calibrate it by rescaling its output. We then estimate $p(y_{\text{ref}}|x)$ and $p(y_{\text{shift}}|x)$ by applying a sigmoid to its output, from which we can estimate $p_{\text{ref}}/p_{\text{shift}}$. To estimate this ratio for the entire shifted dataset, we split the dataset into 10 folds and train a classifier to estimate $p_{\text{ref}}/p_{\text{shift}}$ on each fold using the remaining 9 folds.

Calibrating the classifiers used for splitting As discussed in Appendix F.3, our method for dividing a shifted dataset into an in-support split and an out-of-support split requires a *calibrated* classifier to distinguish between examples from the reference and shifted datasets. Recall that to distinguish between examples from the reference and shifted datasets, we fine-tune a CLIP ViT-L/14 pre-trained on LAION-2B from OpenCLIP. Such over-parameterized models can be overconfident in their predictions (and thus uncalibrated), so we calibrate the classifier by rescaling its (logit) output.

In particular, let f be a (potentially uncalibrated) classifier trained to distinguish between examples from the reference and shifted datasets (where the output of f is a logit). We find the scaling parameter α that minimizes the standard logistic loss of f on a calibration set S_{cal}:

$$\alpha = \arg\min_{\alpha'} \sum_{(x,y)\in S_{\text{cal}}} \log(1+e^{-\alpha'\cdot f(x)\cdot y}). \tag{7}$$

We then define a rescaled classifier $f_{\text{cal}}(x)=\alpha\cdot f(x)$ (which is used to estimate the ratio $p_{\text{ref}}/p_{\text{shift}}$). We produce calibration curves of the rescaled classifiers for each of the shifted datasets we split (see Figure 11) and observe that they are indeed well-calibrated.

F.3.2 Specifications of ImageNet models

To measure the robustness benefits of pre-training on in-support and out-of-support splits of ImageNet distribution shifts, we take existing models from PyTorch Image Models [55]. We establish a baseline for robustness by taking 77 models of different architectures trained from scratch on ImageNet. We compute the effective robustness of 24 pre-trained models that are fine-tuned on ImageNet. We select only models pre-trained on large web-scale datasets such as LAION-2B [3] and IG-1B [59], as these models are likely to have higher effective robustness.

F.4 Curating datasets for fine-tuning

Image editing to synthesize "counterfactual examples" In order to curate a "de-biased" dataset for hair color classification, we edit images from CelebA-HQ [60], a subset of the CelebA dataset with segmentation masks for each attribute provided by CelebAMask-HQ [61]. To change the hair color in a given image, we use InstructPix2Pix [62], a recent image editing model fine-tuned from Stable Diffusion [63]. This model accepts an input image to be edited along with a prompt describing the desired change (e.g., "change the hair color to blond"). We find that InstructPix2Pix is able to

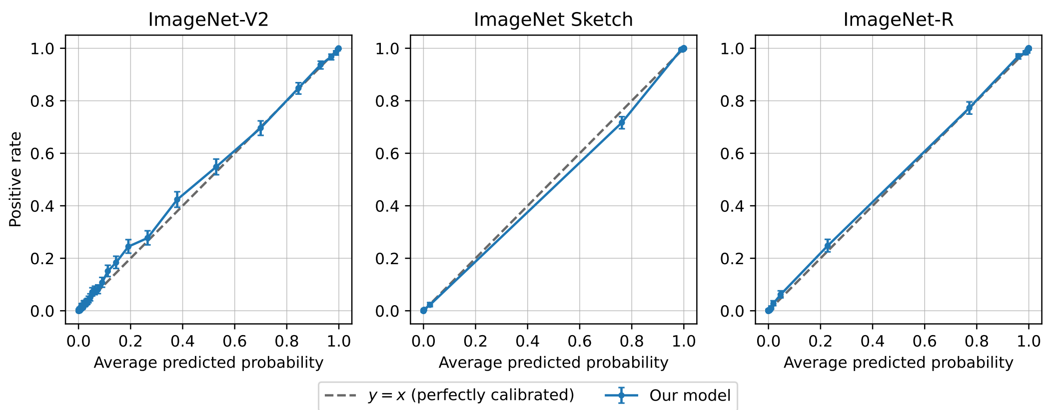


Figure 11: **Calibration curves of classifiers used for splitting.** We display calibration curves for the classifiers used to divide ImageNet-V2, ImageNet-Sketch and ImageNet-R into in-support and out-of-support splits. Specifically, we sort the outputs of each classifier on a combined dataset of reference and shifted examples into 100 bins (where bin edges are quantiles). For each bin, we compute the actual positive rate (i.e., the proportion of examples from the shifted dataset) and the average predicted probability of an example being from the shifted dataset. When we plot the actual positive rates against average predicted probabilities, they are close to equal (close to $y = x$), suggesting that the classifiers are well-calibrated. Error bars denote 95% Clopper-Pearson confidence intervals.

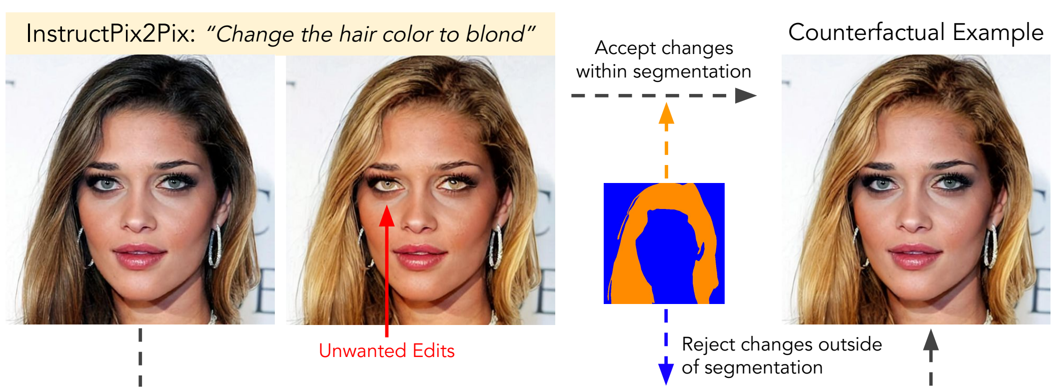


Figure 12: **Synthesizing counterfactual examples.** We edit hair color in CelebA-HQ images using InstructPix2Pix [62]. However, this model can also make unwanted changes to attribute other than hair color, e.g., changing eye color (left). To avoid such issues, in the final image we incorporate only changes within the hair region of the image.

successfully edit the hair color; however, this model often makes undesired changes to attributes such as skin tone and eye color (see, e.g., the left side of Figure 12). To ensure that we only edit hair color, we use the attribute masks to isolate the pixels in a given image corresponding to the hair region, and ignore any changes made outside of this area. When using a binary mask, this procedure could cause unnatural "edges" along the border of the mask. Thus, we apply a Gaussian blur to the hair mask to smooth the transition when "merging" the original and edited images.

To edit an image from non-blond to blond, we use the prompt "change the hair color to blond." When editing from blond to non-blond, however, we find that the prompt "change the hair color to non-blond" gives inconsistent results, likely because the instruction is vague. We observe that most non-blond people in the CelebA dataset have brown or black hair, so as a simple heuristic we randomly edit each image with either the prompt "change the hair color to brown" or the prompt "change the hair color to black." See Figure 12 for a visualization of the image editing process.

Shared model specifications. Accuracy and worst-group accuracy on the CelebA dataset are sensitive to hyperparameter choices. As a result, we conduct a grid search to select hyperparameters for each type of model. We use class-balanced accuracy as the metric for hyperparameter selection, which empirically better correlates with worst-group accuracy than standard accuracy.

When selecting hyperparameters for a curated dataset of a given size, we randomly sample 32 datasets of that size from a pool of $16,000$ images (i.e., $8,000$ CelebA images and their corresponding counterfactual synthesized images) and average the class-balanced accuracies of models trained on each dataset. When evaluating the accuracy and worst-group accuracy of models trained on a curated dataset of a given size, we similarly randomly sample 64 datasets of that size and report average metrics.

For all models, we use the FFCV implementation of *RandomHorizontalFlip* for data augmentation.

Specifications of models trained from scratch. We train ResNet-18 models from scratch by running SGD for 32 epochs, using a triangular learning rate schedule with 4 warmup epochs. We use a batch size of 128, a weight decay of 5×10^{-4} and a momentum of 0.9. We select the best combination of batch size and learning rate from batch sizes of $64, 128, 256, 512$ and learning rates of $0.5, 0.2, 0.1, 0.05, 0.02, 0.01$.

When training models from scratch on our curated dataset, we run SGD for 512 epochs and use a triangular learning rate schedule with 64 warmup epochs. We use a batch size equal to the total number of examples when it is less than 512 and a batch size of 512 otherwise. We use a weight decay of 5×10^{-4} and a momentum of 0.9. We select the best learning rate from $0.5, 0.2, 0.1, 0.05, 0.02, 0.01$.

Baseline specifications. To establish a baseline, we train 100 models from scratch on subsets ranging from 5% of the dataset to the entire dataset.

Specifications of pre-trained models. The pre-trained model in this experiment is a CLIP ViT-B/32 model initialized as a zero-shot classifier with "blond" and "non-blond" as the class names. We fine-tune models by running AdamW for 16 epochs, using a cosine learning rate schedule with 2 warmup epochs, and a weight decay of 0.1. We select the best combination of batch size and learning rate from batch sizes of $64, 128, 256, 512$ and learning rates of $3 \times 10^{-5}, 1 \times 10^{-5}, 3 \times 10^{-6}, 1 \times 10^{-6}$.

When training on our curated dataset, we use a batch size of 64 (the size of the dataset) and select the best learning rate from $3 \times 10^{-5}, 1 \times 10^{-5}, 3 \times 10^{-6}, 1 \times 10^{-6}$.