Dataset Interfaces: Diagnosing Model Failures Using Controllable Counterfactual Generation

Joshua Vendrow [*1] **Saachi Jain** [*1] **Logan Engstrom** [1] **Aleksander Madry** [1]

Abstract

Distribution shift is a major source of failure for machine learning models. However, evaluating model reliability under distribution shift can be challenging, especially since it may be difficult to acquire *counterfactual examples* that exhibit a specified shift. In this work, we introduce the notion of a *dataset interface*: a framework that, given an input dataset and a user-specified shift, returns instances from that input distribution that exhibit the desired shift. We study a number of natural implementations for such an interface, and find that they often introduce confounding shifts that complicate model evaluation. Motivated by this, we propose a new implementation that leverages Textual Inversion to tailor generation to the input distribution. We then demonstrate how applying this dataset interface to the ImageNet dataset enables studying model behavior across a diverse array of distribution shifts, including variations in background, lighting, and attributes of the objects.

1. Introduction

Suppose we would like to deploy a vision model (for example, one trained on ImageNet). Naturally, we would like this model to perform reliably in a variety of real-world contexts and, especially, with respect to any of the (inevitable) corner cases, i.e., real-world inputs that are underrepresented in the training dataset. Indeed, we have ample evidence that machine learning models can fail when facing so-called distribution shift, including changes of the background (Beery et al., 2018; Xiao et al., 2020; Barbu et al., 2019) and object pose (Engstrom et al., 2019; Alcorn et al., 2019), as well as variability in data collection pipelines (Recht et al., 2019; Engstrom et al., 2020).

How can we then ensure that our model will continue to perform well in new environments? To make this more concrete, suppose that our hypothetical ImageNet-trained model is being deployed to identify objects (such as dogs, chairs, plates, etc.) in images. We would like to make sure that the model will perform well regardless of the object's type (e.g., "brown dog", "ceramic plate"), condition (e.g., "dog with harness", "empty plate"), and setting (e.g., "dog on a beach", "plate at a picnic"). One powerful primitive for assessing our model's performance in such scenarios is *counterfactual generation* — acquiring images (*counterfactual examples*) that conform to the training distribution except for exhibiting a specified change. For instance, to test the model's performance on "plate with utensils," we might want to evaluate our model on images of plates that match the distribution of ImageNet (e.g., in terms of background, zoom, plate style) except that they have utensils alongside them. But how can we acquire such counterfactual examples? After all, without access to the original data-generating process, collecting new examples in a specified context can be challenging.

Currently, practitioners use a few natural methods for generating counterfactual examples. Returning to our example of "plate with utensils," one (labor intensive) strategy is to manually take photographs of ImageNet-style scenes of plates with and without utensils. More scalable approaches include scraping images of plates with utensils from the internet (e.g., using Google or Bing search engines), or synthetically generating such images (e.g., using a text-to-image diffusion model such as Stable Diffusion (Rombach et al., 2022)). However, images produced using such approaches often reflect additional *confounding shifts*. For example, querying Bing or Stable Diffusion with the prompt "a photo of a plate with utensils" surfaces plates that are almost exclusively empty, while in ImageNet the plates usually contain food (see Figure 1). Any observed change in the model's accuracy on these images might thus well be due to this confounding shift rather than the shift of interest (i.e., the presence of utensils).

Our Contributions

In this work, we unify approaches to counterfactual generation under a common notion of a *dataset interface*: a

[*]Equal contribution [1]Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA. Correspondence to: Joshua Vendrow <jvendrow@mit.edu>.

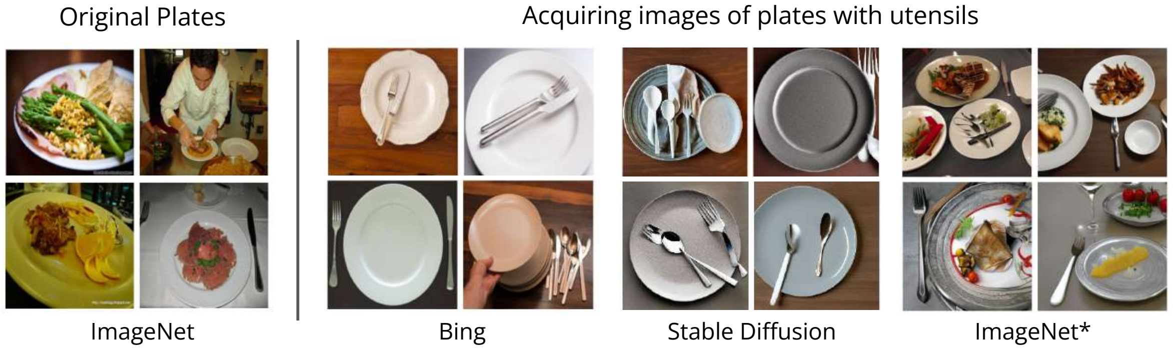


Figure 1. Acquiring ImageNet counterfactual examples of "plates with utensils" using the Bing search engine, Stable Diffusion, or our dataset interface ImageNet∗). The Bing and Stable Diffusion plates are almost exclusively empty, and thus do not fully match the images in ImageNet (as the latter often contain food). In contrast, ImageNet∗ can generate counterfactual examples of "plates with utensils" that match the ImageNet distribution much more closely.

primitive that, given an input dataset and a user-specified shift, aims to return instances from the input distribution that exhibit the desired shift. We then study a number of existing strategies for implementing such an interface and find that — due to a mismatch between the distribution produced by the interface and that of the input dataset — these approaches often introduce confounding shifts that can complicate model evaluation.

To mitigate this mismatch, we introduce a new implementation of a dataset interface that leverages Textual Inversion (Gal et al., 2022) with Stable Diffusion. In particular, this implementation encodes each class in the input dataset as a token within the text-space of the diffusion model. By integrating these tokens into natural language prompts, our implementation can generate counterfactual examples that conform to the input distribution while still exhibiting the desired shift. Overall, this implementation:

- **Is tailored to the input dataset:** It can match key aspects of the original dataset, even for objects and attributes without a clear textual specification. For example, if the input dataset contains a specific breed of dog, our dataset interface can generate images matching that breed, even if the underlying diffusion model is unable to associate this breed with any natural language description.
- **Provides fine-grained control:** It can generate images of a target object with a high level of control over the desired distribution shift. This includes manipulating not only aspects such as backgrounds (e.g., "on a beach") and lighting (e.g., "in studio lighting"), but also more fine-grained adjustments such as co-occurring objects (e.g., "with a person") and attributes of the objects themselves (e.g., "lying down").
- **Enables scalable counterfactual generation:** It is able to rapidly generate counterfactual examples, allowing us to evaluate a model's robustness across many possible failure modes.

Finally, leveraging this implementation, we create ImageNet∗, a dataset interface for the ImageNet dataset (Russakovsky et al., 2015). We then demonstrate how we can use this interface to evaluate the performance of ImageNet-trained models under a diverse array of distribution shifts. In particular, due to our implementation's scalability and flexibility, we can use our interface to further a *shift-centric* perspective on model robustness, by categorizing how performance on *different* types of shifts scales with model size, architecture, and pre-training regime.

2. Dataset Interfaces: Unifying methods for Counterfactual Generation

Let us return to our example of deploying a vision model for object classification. Suppose we wish to evaluate that model's ability to correctly classify images of a dog in a variety of contexts (e.g., "on a beach"). To do so, we would like to collect *counterfactual examples*, i.e., examples that exhibit the required distribution shift ("on a beach") but still contain the original object ("dog") as it appears in the training distribution.

In order to unify such strategies for collecting counterfactual examples, we introduce the notion of a *dataset interface*: a primitive that, given an input dataset and a user-specified shift, aims to return instances of a class from that dataset that exhibit the desired shift. In general, users do not have access to the original data-generating process, making it difficult to retrieve new examples with a specified context. A dataset interface thus serves as a proxy for the original dataset that enables users to control (and edit) the desired aspects of the surfaced images.

What makes a good dataset interface? In order to facilitate wide-scale model evaluation, a dataset interface needs

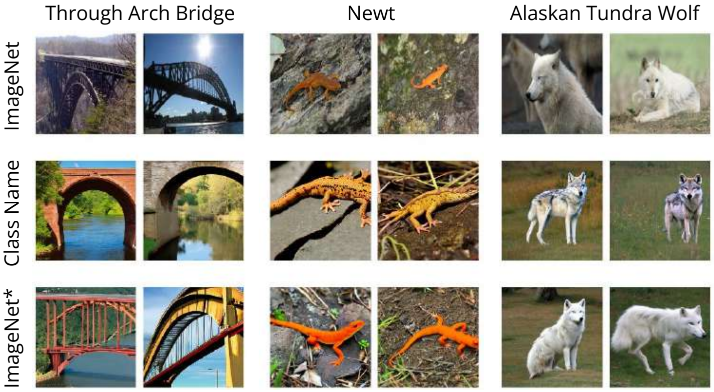


Figure 2. Examples of real images from ImageNet (**top**) and images generated either by prompting Stable Diffusion with the class name (e.g., "a photo of a newt") (**middle**) or by using our dataset interface ImageNet∗ (**bottom**). For each class, the images from ImageNet∗ match ImageNet more closely than the images generated using the class name (see, e.g., the columns in the bridges). See Appendix B for further examples.

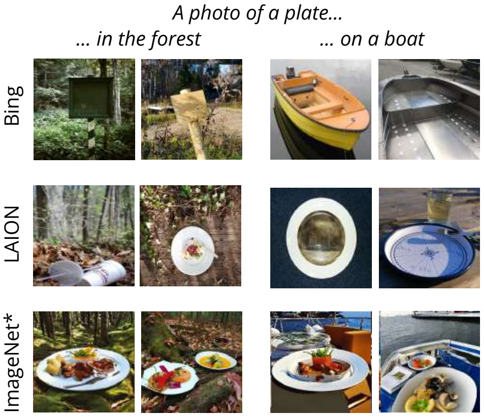


Figure 3. Examples of plates in various contexts collected from Bing (**top**), LAION (**middle**), or generated using our dataset interface ImageNet∗ (**bottom**). Plates queried via text-to-image retrieval often miss either the object or shift, while the ImageNet∗ plates contain both.

to fulfill three criteria: (1) images returned by the interface should exhibit the desired shift; (2) images returned by the interface should contain the specified object (as it appears in the input distribution); and (3) the interface needs to return images quickly in order to scale to many classes and contexts. In the following section, we discuss how existing approaches perform under these three criteria.

2.1. Existing Instantiations of Dataset Interfaces

Currently, practitioners use a variety of techniques for surfacing counterfactual examples. Here, we discuss three categories of existing approaches — manual data collection, text-to-image retrieval, and synthetic data generation — that fit into the dataset interface framework. As a running example, let's return to our example of evaluating a "dog" classifier on a variety of backgrounds (e.g., "a beach").

Manual data collection Perhaps the most straightforward strategy (as implemented in ObjectNet (Barbu et al., 2019), iWildCam (Beery et al., 2018)) for acquiring counterfactual examples is to manually collect them from the real world. In our example, we could find the same types of dogs that appeared in our original dataset, and then take photos of those dogs on different backgrounds. This kind of data collection gives practitioners strong control over what the images they collect contain, but can be very expensive and time-consuming (and, for rarer objects like "polar bear", may be infeasible).

Text-to-image retrieval A more common (and far more scalable) method is to find images from the internet using a textual query that matches our desired context (e.g., "a photo of a dog on a beach"). One such approach (as implemented in ImageNet-R (Hendrycks et al., 2020), ImageNet-Sketch (Wang et al., 2019)) is to query an image search engine like Bing or Google. Another strategy (as implemented in ADAVISION (Gao et al., 2022)) is to retrieve images from a (huge) dataset of text-image pairs (e.g., LAION-5B (Schuhmann et al., 2022)). For example, `clip-retrieval` (Beaumont, 2022) uses a KNN index on top of a CLIP (Radford et al., 2021) latent space — a joint language-image embedding space learned with a contrastive objective — to return images corresponding to a textual prompt. However, such text-to-image retrieval methods typically assume that a valid counterfactual example exists within the provided pool of images. We find that for many uncommon examples these methods are often unable to find a valid counterfactual example that includes both the desired object and shift (see Figure 3).

Synthetically generated data Synthetic generation provides a more flexible mechanism for generating images in rarer contexts. Until recently, synthetic counterfactual examples were generated either by pre-processing images to induce distribution shifts (e.g., ImageNet-C (Hendrycks & Dietterich, 2019)), or rendering scenes using a 3D simulator (Hamdi et al., 2018; Alcorn et al., 2019; Hamdi & Ghanem, 2019; Shu et al., 2020; Jain et al., 2020; Leclerc et al., 2021). However, these methods often produce images that are not photorealistic, or require involved processing steps (e.g., collecting a 3D scan of an object). More recently, taking advantage of current progress in generative models, other works (Kattakinda et al., 2022; Wiles et al., 2022) employ off-the-shelf text-to-image models such as Stable Diffusion (Rombach et al., 2022), DALL-E 2 (Ramesh et al., 2022), and Imagen (Saharia et al., 2022) to generate photorealistic images conditioned on a textual prompt. In our running example, we could prompt Stable Diffusion to gen-

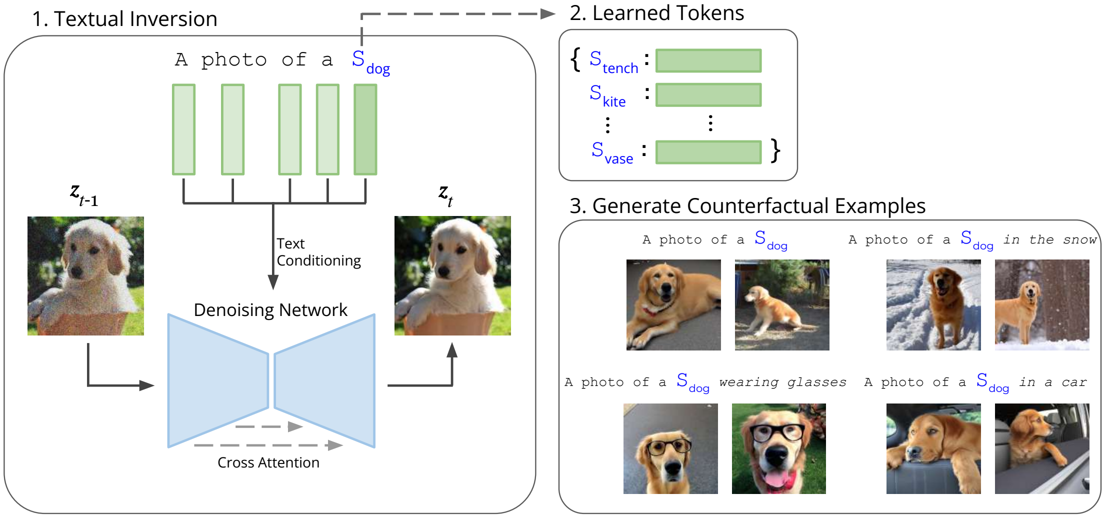


Figure 4. Construction of our dataset interface. For each class in the input dataset, we use Textual Inversion (see Section 3) to learn a token in the text space of a text-to-image diffusion model. This token is intended to capture the distribution of the corresponding class. Then, by incorporating these tokens in natural language prompts, we can scalably generate a dataset of counterfactual examples.

erate dogs on different backgrounds using a corresponding textual query (such as “a photo of a dog on a beach.”)

However, synthetic generation can often introduce biases that result in confounding shifts. For example, suppose that the input dataset only contains certain breeds of dogs. In this case, the diffusion model’s conception of “dog” could differ from the dogs in the “input” dataset (e.g., a completely different set of breeds). As a result, simply employing prompts that use the class name might not be sufficient to faithfully match the distribution of that dataset. In fact, we find that for a number of classes in the ImageNet dataset, there is a visual discrepancy between images generated by Stable Diffusion using the class name (e.g., “dog”) and the images in that dataset itself. For instance, as we show in Figure 2, there might be a mismatch in the specific type or appearance of the object (e.g., a subspecies of wolf, or columns in bridges). How can we then generate images that faithfully correspond to the distribution of the input dataset?

3. Generating dataset-specific counterfactual examples

To overcome this mismatch, we propose an implementation of a dataset interface that bridges the gap between the dataset interface and the corresponding input dataset. Specifically, we leverage recent work in *personalized* text-to-image generation, which tries to incorporate user-provided visual concepts within a text-to-image diffusion model. By doing so, we can generate images that faithfully capture the properties of the corresponding class in the input dataset and avoid confounding shifts. In this work, we use Textual Inversion (Gal et al., 2022) (although it is possible to implement a similar interface with other methods for personalized generation such as DreamBooth (Ruiz et al., 2022)).

Textual Inversion Given a set of user-provided images containing a desired visual concept, Textual Inversion aims to find a “word” (token) S_* in the diffusion model’s text space to precisely capture that concept. This token can then be included in natural language prompts to generate images incorporating the desired concept. So, for example, using the prompt “a monochrome photo of a S_*” should result in generating black and white images with that concept.

In order to create such a customized token S_*, Textual Inversion learns a corresponding embedding vector v_* in the text embedding space of the diffusion model. To learn this embedding vector v_*, Textual Inversion freezes the weights of the pre-trained diffusion model and then finds v_* that minimizes the diffusion model’s original training objective, while using only the user-provided images that capture the desired visual concept paired with prompts containing S_* (e.g., “a photo of a S_*”).

Encoding the input dataset as tokens in text space With Textual Inversion in hand, we aim to guide a text-to-image diffusion model to generate images more closely aligned with the objects in the input dataset. Specifically, for each class c from that dataset, we run Textual Inversion on the training images of that class to learn an embedding vector v_c for a corresponding new class token S_c. We can then incorporate these class tokens into our prompts to generate images under our desired shift. For example, to generate an image of a dog on the beach, we can use the prompt “A photo of a S_{dog} on the beach.” We present an overview of our construction in Figure 4.

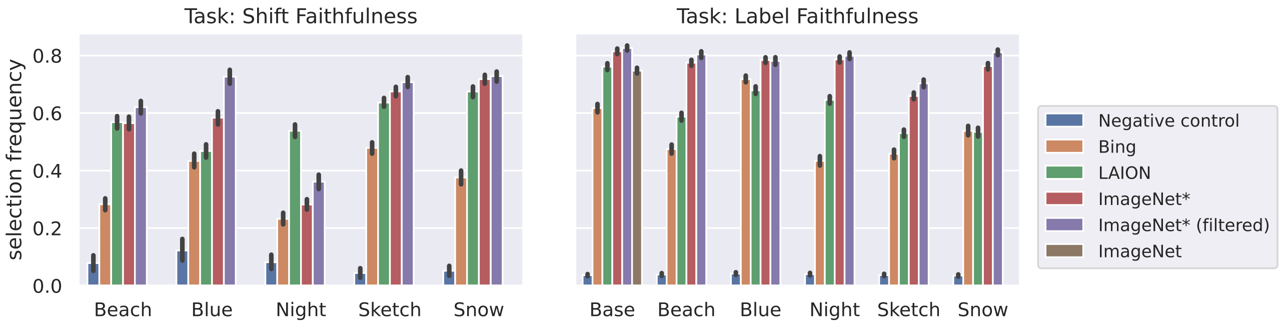


Figure 5. Selection frequencies (see Section 4) for images from ImageNet, images scraped using Bing, image retrieval from LAION, and images generated with ImageNet$*$ when asking workers to identify either presence of a specific distribution shift (**left**) or presence of a ImageNet class (**right**) in the image. The ImageNet$*$ images exhibit the desired context and object more often than those from both Bing and LAION. Filtering the images with the CLIP metrics consistently increases selection frequency for both the shift and the object.

3.1. Controlling the quality of counterfactual examples

Text-to-image diffusion models can sometimes make mistakes, and as a result some of the counterfactual examples generated by our text prompts might either (1) not contain the original object or (2) not depict the desired distribution shift. We thus leverage CLIP (Radford et al., 2021) to control the quality of "candidate" counterfactual examples based on these two criteria. Specifically, given a text label $<$*class*$>$ of a class (e.g., "dog") and a text description $<$*shift*$>$ of the desired distribution shift (e.g., "on the beach"), we construct the captions c_{class} = "a photo of a <class>" and c_{shift} = "a photo <shift>."

Then, to quantify the presence of the original object and the desired distribution shift within an image, we measure the *CLIP similarity*, i.e., the similarity between the CLIP embedding of the image and the text embeddings of captions c_{class} and c_{shift} respectively[1]. We use these metrics to automatically filter and remove images that do not meet the above criteria (see Appendix A.3 for details, and Appendix C for yield rates.) A user study in Section 4 confirms that this filtering step indeed improves the quality of the resulting dataset of counterfactual examples.

3.2. ImageNet$*$

We apply the construction described above to create ImageNet$*$, a dataset interface for the ImageNet dataset (we defer results for other datasets to Appendix D) and we publicly release the resulting set of 1,000 learned class tokens.

In Section 6, we will use ImageNet$*$ to create a distribution shift robustness benchmark consisting of counterfactual examples for 23 different distribution shifts, including shifts in background, lighting, style, and object co-occurrence. We publicly release this benchmark as well, but we also encourage users to generate their own counterfactual examples tailored to their specific needs.

4. Evaluating the Generated Counterfactual Examples

Having constructed ImageNet$*$ (see Section 3.2), we now evaluate the quality of our generated images. Specifically, we use ImageNet$*$ to synthesize images of distribution shifts from five different categories — "at night", "blue", "in the beach","in the snow", and "sketch" — as well as "base" images which do not correspond to a specific shift (generated with a prompt "a photo of a S*"). As a baseline, we consider downloading images returned by the Bing search engine or retrieving them with `clip-retrieval`, an open source tool for scraping LAION, when queried with a natural language prompt containing the ImageNet class name. (See Appendix A.2 for experimental details.)

We validate the quality of these images through a user study on the Amazon Mechanical Turk (MTurk) crowdsourcing platform. In this study, we show workers a grid of images, either sampled from ImageNet$*$, scraped from the Bing engine, or retrieved from LAION-5B with `clip-retrieval`, with additional images from ImageNet as a control. We then ask the workers to identify which images contain (a) the target ImageNet class (e.g., "golden retriever") and (b) the desired distribution shift (e.g., "on the beach"). See Appendix A.4 for further details.

In Figure 5, we report the *selection frequency*, i.e., the fraction of images selected by the workers, for each of these two tasks. We find that the images generated by our framework exhibit the desired shift and object of interest more often than images scraped using Bing across each distribution shift. Querying LAION with `clip-retrieval` is

[1] For shifts describing art styles (e.g., "a sketch of a ...") we instead use c_{class} = *"a <class>"* as they are no longer a "photo."

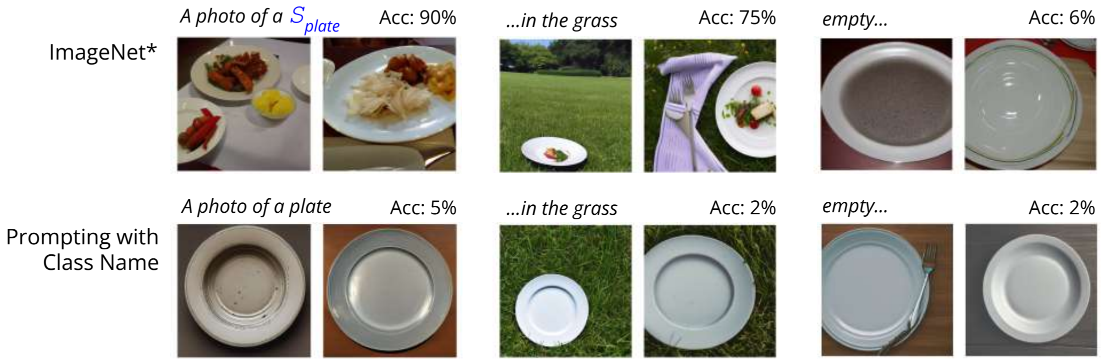


Figure 6. Images of plates generated by our dataset interface ImageNet∗ (**top**) and by prompting Stable Diffusion with the class name "plate" (**bottom**). While generating ImageNet∗ examples "in the grass" causes only a slight drop in accuracy, prompting Stable Diffusion with *"a plate in the grass"* degrades the model's accuracy to 2% due to the additional confounding factor of emptiness.

competitive with ImageNet∗ in retrieving images with the desired shift, but struggles with keeping the desired object. We also observe that filtering the generated images with our metrics improves the selection frequency for both tasks.

5. Fine-Grained Model Debugging

Capturing certain class-specific model failures may require executing fine-grained adjustments in particular scenarios (such as adding a harness on a dog or putting a fish in a tank). Our dataset interface provides exactly this kind of debugging capability while avoiding the unintended effect of secondary "confounding" shifts. To illustrate this, let us return to our example task of deploying an ImageNet-trained model. Suppose that we would like to examine the model's performance on images of plates in the grass. When prompting Stable Diffusion with a query that uses the class name ("a photo of a plate in the grass"), we find that our classifier achieves an accuracy of *only 2%* on the resulting images! Is the grassy background really such a catastrophic failure mode for our model?

If, instead, we use ImageNet∗ to generate counterfactual examples of "plates on the grass", we find that our classifier's accuracy only slightly drops from 90% to 75% (see Figure 6). What causes this discrepancy? It turns out that the "failure case" identified when prompting Stable Diffusion with natural language is actually an extreme example of a confounding shift. Indeed, recall that ImageNet plates usually have food on them (c.f., Figure 1). However, the plates generated by Stable Diffusion are almost exclusively empty, even when using a prompt that in principle does not introduce any shift (i.e., "a photo of a plate").

To assess to what degree this confounding shift of "emptiness" is detrimental for our ImageNet classifier, we use ImageNet∗ to generate counterfactual examples of empty plates. We then find that the classifier's accuracy decreases to only 6% (so, similar to the 2% we observed before). To further confirm that the failure mode is indeed caused by emptiness and not the presence of grass, we took real photos of a plate in each of these contexts and evaluated our model on them (see Figure 14).

So, as we have seen, dataset interfaces enable us to test distribution shifts in *isolation*, i.e., without introducing confounding shifts that can produce misleading results. In Appendix B, we discuss additional examples of using our interface for precise model debugging.

6. Evaluating Distribution Shift Robustness

Our framework's scalability enables us to rapidly assess a model's performance on a wide variety of distribution shifts. As a result, we can take a *shift-centric* perspective on robustness by evaluating models on many types of shifts at once and categorizing variations in these models' behavior.

A benchmark for distribution shift robustness. Using ImageNet∗, we generate images for 23 shifts, including changes in background, weather, lighting, style, attributes, and co-occurrence (see Figure 7 for examples, and Appendix A.5 for a full list). We then evaluate a variety of image classification models varying architectures, training regimes, pre-training schemes, and input resolutions.

We can now categorize the behavior of each shift according to two criteria. The first criteria, the shift's *absolute impact*, encapsulates the shift's overall severity, and can be measured as the average difference between the models' performance on the base generated images and the corresponding counterfactual examples. The second criteria, the *ID/OOD slope* captures the degree to which improving model accuracy on in-distribution images also boosts its performance under the distribution shift. We measure this quantity by plotting the accuracy of each model on the base generated images versus on the counterfactual examples (as in (Taori et al., 2020;

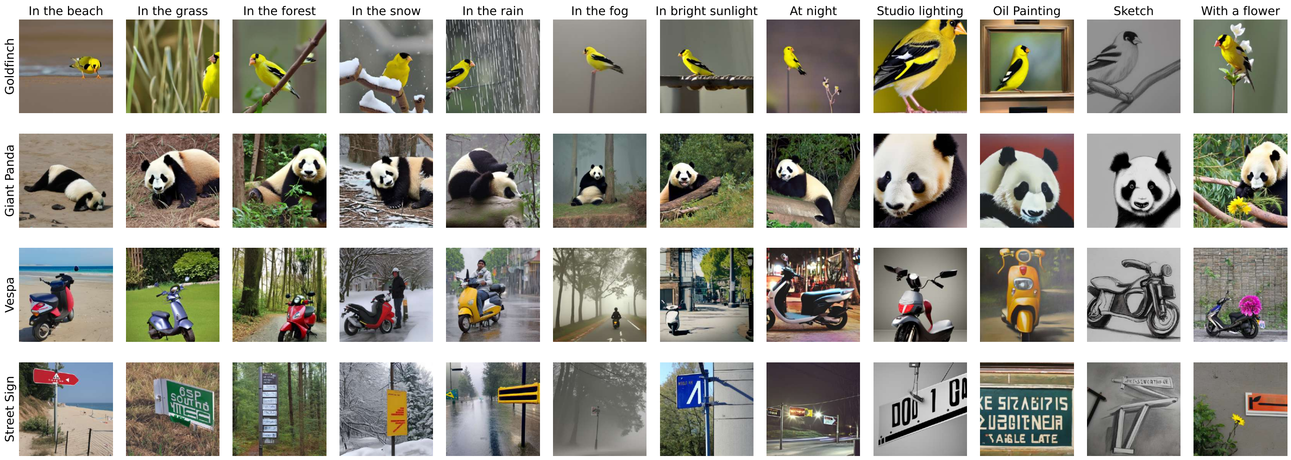


Figure 7. Examples of images generated with ImageNet∗ for a variety of distribution shifts. These images are a subset of the benchmark described in Section 6. See Appendix B for further examples.

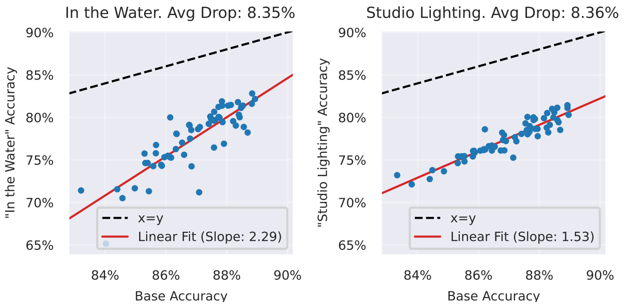


Figure 8. Accuracy on images generated with a distribution shift vs. on images generated with the base prompt (i.e., "a photo of S_*") for two distribution shifts over a sweep of ImageNet models.

Miller et al., 2021)), and then calculating the slope of the best-fit-line (see Figure 8 for two examples).

In Figure 9, we plot the absolute impact and the ID/OOD slope for each one of the considered distribution shifts. We find that different types of shifts result in different scaling behaviors across these two criteria. For example, even though "in the water" and "studio lighting" have similar absolute impacts, "in the water" has a higher ID/OOD slope. Therefore, while boosting the in-distribution accuracy for ImageNet can help improve the model's performance on images "in the water", the model's performance on "studio lighting" is much more static. More broadly, we find that shifts based on lighting (e.g., "studio lighting") have lower ID/OOD slope than shifts based on background (e.g., "in the grass"), with attributes (e.g., "red") in between.

7. Related Work

Benchmarks for distribution shift robustness Many robustness benchmarks evaluate model performance under specific distribution shifts by collecting real images. These include shifts in style (Hendrycks et al., 2020; Wang et al., 2019), object pose (Barbu et al., 2019), background (Beery et al., 2018), time/location (Christie et al., 2018; Hendrycks et al., 2020), and data pipelines (Recht et al., 2019). Other works create synthetic distribution shift benchmarks, often by preprocessing the images of an "in-distribution" dataset to induce a shift. One common strategy here is to synthetically create shifts in background by pasting the foreground of the target image onto an alternate background image (Xiao et al., 2020; Sagawa et al., 2020; Kattakinda et al., 2022). In particular, ImageNet-C (Hendrycks & Dietterich, 2019) applies a set of transformations such as blur and synthetic fog on top of images to simulate real-world corruptions. Finally, TILO (Lynch et al., 2022) uses Stable Diffusion to generate images of vehicles with variations in backgrounds and lighting.

Identification of failure modes through counterfactual examples There are a number of works that aim to diagnose model failures by evaluating them on counterfactual examples. One line of such work leverages 3D rendering software to synthesize objects with varying geometry and pose (Hamdi et al., 2018; Alcorn et al., 2019; Hamdi & Ghanem, 2019; Shu et al., 2020; Jain et al., 2020; Leclerc et al., 2021). Of these, 3DB (Leclerc et al., 2021) is the closest to our work, as in addition to pose, they allow control over aspects such as lighting, background, texture, and object co-occurrence. However, their framework still requires users to first acquire a 3D model of the object of interest.

On the other hand, ADAVISION (Gao et al., 2022) introduces an interactive process for identifying model failures by repeatedly querying for real images from LAION-5B and optimizing the query to more closely match the model's misclassifications. However, ADAVISION requires user

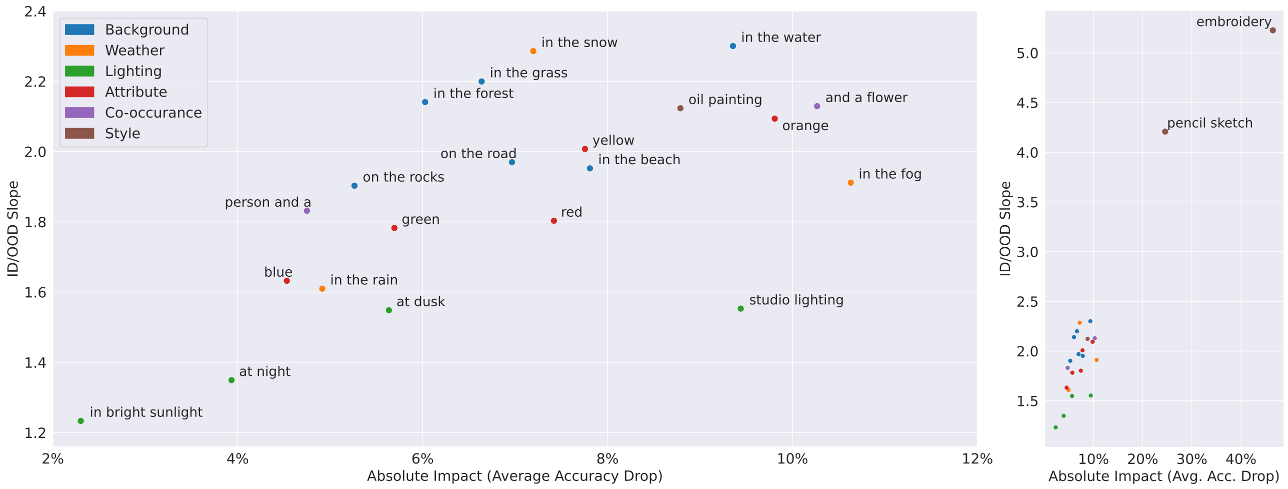


Figure 9. For each of the shifts in the benchmark, we plot the ID/OOD slope (degree to which in-distribution accuracy improves accuracy under the shift) versus absolute impact (average drop in accuracy due to the shift). The "'pencil sketch" and "embroidery" shifts are shown separately on the right, as their absolute impact and ID/OOD slope fall far from all other shifts (points for the other shifts are shown as reference). Note that changes in lighting (e.g., "at dusk") and attribute (e.g., "red") have low ID/OOD slope, indicating that the models' performance on these images remains static regardless of in-distribution accuracy.

intervention at each step to verify model failures. Wiles et al. (2022), in turn, propose a framework for automatically surfacing model failures by generating images with a text-to-image generative model, clustering misclassified inputs, and then using a image-to-text model to caption these clusters. Finally, Jain et al. (2022) synthesize prototypical examples of challenging subpopulations by automatically captioning model failure modes and then generating the images via Stable Diffusion.

Personalized text-to-image generation While our dataset interface utilizes textual inversion to learn personalized concepts, there have been many recent works in the context of personalized text-to-image generation. These approaches aim to incorporate user-provided visual concepts (e.g., an object or style) into text-to-image generation. One family of techniques uses a guiding image to further condition generation (Jeanneret et al., 2022; Yuan et al., 2022; Kattakinda et al., 2022). Specifically, D3S (Kattakinda et al., 2022) first pastes the foreground of a given object onto a background and then uses the resulting image to guide Stable Diffusion. Yuan et al. (2022) generate images in a desired target domain by conditioning on an image from the source dataset and a prompt that describes the target domain.

Another approach to personalized generalization, taken by methods such as Textual Inversion (Gal et al., 2022) and DreamBooth (Ruiz et al., 2022), allows users to directly encode a desired concept within the text space of the text-to-image model. While Textual Inversion learns a new token within a frozen text-to-image model, DreamBooth fine-tunes the full model. By allowing the original diffusion model weights to change, DreamBooth offers greater capability for personalization at the potential cost of degrading the generation of concepts already known to the model.

8. Conclusion

In this work, we introduce the notion of a dataset interface: a framework that, given an input dataset and user-specified shift, returns instances from that input distribution that exhibit the desired shift. While there are a number of existing implementations of such an interface, they often introduce confounding shifts due to a mismatch between the interface and the input dataset. To mitigate this issue, we propose an implementation of a dataset interface that leverages Textual Inversion to tailor counterfactual generation more closely to the input dataset. In addition to enabling fine-grained model debugging, our dataset interface allows users to simultaneously evaluate a diverse array of distribution shifts, making it possible to take a more "shift-centric" perspective on model robustness.

There are several avenues for further investigation. While our dataset interface implementation leverages natural language descriptions to represent distribution shifts, one could instead attempt to automatically learn a representation of a shift using user-provided example images. Another potential direction to explore is to use counterfactual examples generated by such an interface to improve a model's robustness (e.g., by incorporating the generated images into the training pipeline).

References

Alcorn, M. A., Li, Q., Gong, Z., Wang, C., Mai, L., Ku, W.-S., and Nguyen, A. Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Barbu, A., Mayo, D., Alverio, J., Luo, W., Wang, C., Gutfreund, D., Tenenbaum, J., and Katz, B. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In *Neural Information Processing Systems (NeurIPS)*, 2019.

Beaumont, R. Clip retrieval: Easily compute clip embeddings and build a clip retrieval system with them. https://github.com/rom1504/clip-retrieval, 2022.

Beery, S., Van Horn, G., and Perona, P. Recognition in terra incognita. In *European Conference on Computer Vision (ECCV)*, 2018.

Christie, G., Fendley, N., Wilson, J., and Mukherjee, R. Functional map of the world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6 2018.

Engstrom, L., Tran, B., Tsipras, D., Schmidt, L., and Madry, A. Exploring the landscape of spatial robustness. In *International Conference on Machine Learning (ICML)*, 2019.

Engstrom, L., Ilyas, A., Santurkar, S., Tsipras, D., Steinhardt, J., and Madry, A. Identifying statistical bias in dataset replication. In *International Conference on Machine Learning (ICML)*, 2020.

Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A. H., Chechik, G., and Cohen-Or, D. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022.

Gao, I., Ilharco, G., Lundberg, S., and Ribeiro, M. T. Adaptive testing of computer vision models. *arXiv preprint arXiv:2212.02774*, 2022.

Hamdi, A. and Ghanem, B. Towards analyzing semantic robustness of deep neural networks. *arXiv preprint arXiv:1904.04621*, 2019.

Hamdi, A., Muller, M., and Ghanem, B. Sada: Semantic adversarial diagnostic attacks for autonomous applications. *arXiv preprint arXiv:1812.02132*, 2018.

Hendrycks, D. and Dietterich, T. G. Benchmarking neural network robustness to common corruptions and surface variations. In *International Conference on Learning Representations (ICLR)*, 2019.

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., and Gilmer, J. The many faces of robustness: A critical analysis of out-of-distribution generalization, 2020.

Jain, L., Chandrasekaran, V., Jang, U., Wu, W., Lee, A., Yan, A., Chen, S., Jha, S., and Seshia, S. A. Analyzing and improving neural networks by generating semantic counterexamples through differentiable rendering. *arXiv preprint arXiv:1910.00727*, 2020.

Jain, S., Lawrence, H., Moitra, A., and Madry, A. Distilling model failures as directions in latent space. *arXiv preprint arXiv:2206.14754*, 2022.

Jeanneret, G., Simon, L., and Jurie, F. Diffusion models for counterfactual explanations. In *Proceedings of the Asian Conference on Computer Vision*, pp. 858–876, 2022.

Kattakinda, P., Levine, A., and Feizi, S. Invariant learning via diffusion dreamed distribution shifts. *arXiv preprint arXiv:2211.10370*, 2022.

Leclerc, G., Salman, H., Ilyas, A., Vemprala, S., Engstrom, L., Vineet, V., Xiao, K., Zhang, P., Santurkar, S., Yang, G., et al. 3db: A framework for debugging computer vision models. In *arXiv preprint arXiv:2106.03805*, 2021.

Lynch, A., Kaddour, J., and Silva, R. Evaluating the impact of geometric and statistical skews on out-of-distribution generalization performance. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2022.

Maji, S., Rahtu, E., Kannala, J., Blaschko, M., and Vedaldi, A. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Miller, J. P., Taori, R., Raghunathan, A., Sagawa, S., Koh, P. W., Shankar, V., Liang, P., Carmon, Y., and Schmidt, L. Accuracy on the line: on the strong correlation between out-of-distribution and in-distribution generalization. In *International Conference on Machine Learning*, pp. 7721–7735. PMLR, 2021.

Parkhi, O. M., Vedaldi, A., Zisserman, A., and Jawahar, C. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3498–3505. IEEE, 2012.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *arXiv preprint arXiv:2103.00020*, 2021.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Recht, B., Roelofs, R., Schmidt, L., and Shankar, V. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning (ICML)*, 2019.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., and Aberman, K. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arXiv:2208.12242*, 2022.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. In *International Journal of Computer Vision (IJCV)*, 2015.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *International Conference on Learning Representations*, 2020.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022.

Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al. Laion-5b: An open large-scale dataset for training next generation image-text models. In *arXiv preprint arXiv:2210.08402*, 2022.

Shu, M., Liu, C., Qiu, W., and Yuille, A. Identifying model weakness with adversarial examiner. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

Taori, R., Dave, A., Shankar, V., Carlini, N., Recht, B., and Schmidt, L. Measuring robustness to natural distribution shifts in image classification. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Wang, H., Ge, S., Xing, E. P., and Lipton, Z. C. Learning robust global representations by penalizing local predictive power. *Neural Information Processing Systems (NeurIPS)*, 2019.

Wightman, R. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

Wiles, O., Albuquerque, I., and Gowal, S. Discovering bugs in vision models using off-the-shelf image generation and captioning. *arXiv preprint arXiv:2208.08831*, 2022.

Xiao, K., Engstrom, L., Ilyas, A., and Madry, A. Noise or signal: The role of image backgrounds in object recognition. *arXiv preprint arXiv:2006.09994*, 2020.

Yuan, J., Pinto, F., Davies, A., Gupta, A., and Torr, P. Not just pretty pictures: Text-to-image generators enable interpretable interventions for robust representations. 2022.

A. Setup Details

A.1. Textual Inversion

To learn each token, we run textual inversion with $3,000$ optimization steps. We use an adam optimizer wth a constant learning rate schedule, a learning rate of $5e-4$, $\beta_1 = 0.9$, $\beta_1 = 0.999$, and weight decay $1e-2$. Our hyperparameters follow the HuggingFace textual inversion script at:

https://github.com/huggingface/diffusers/tree/main/examples/textual_inversion.

A.2. Scraping images from Bing and LAION

For the Bing engine baseline, we query Bing with natural language prompts. We leverage the scraping library bing-image-downloader, and scrape the top 50 images per class.

For the LAION baseline, we query LAION with the `clip retrieval` tool (Beaumont, 2022) (see https://rom1504.github.io/clip-retrieval/ for a Web UI). We scrape the top 50 images per class using CLIP VIT-H/14.

A.3. Setting CLIP Thresholds

Here we describe our procedure for setting thresholds our CLIP similarity metric when filtering.

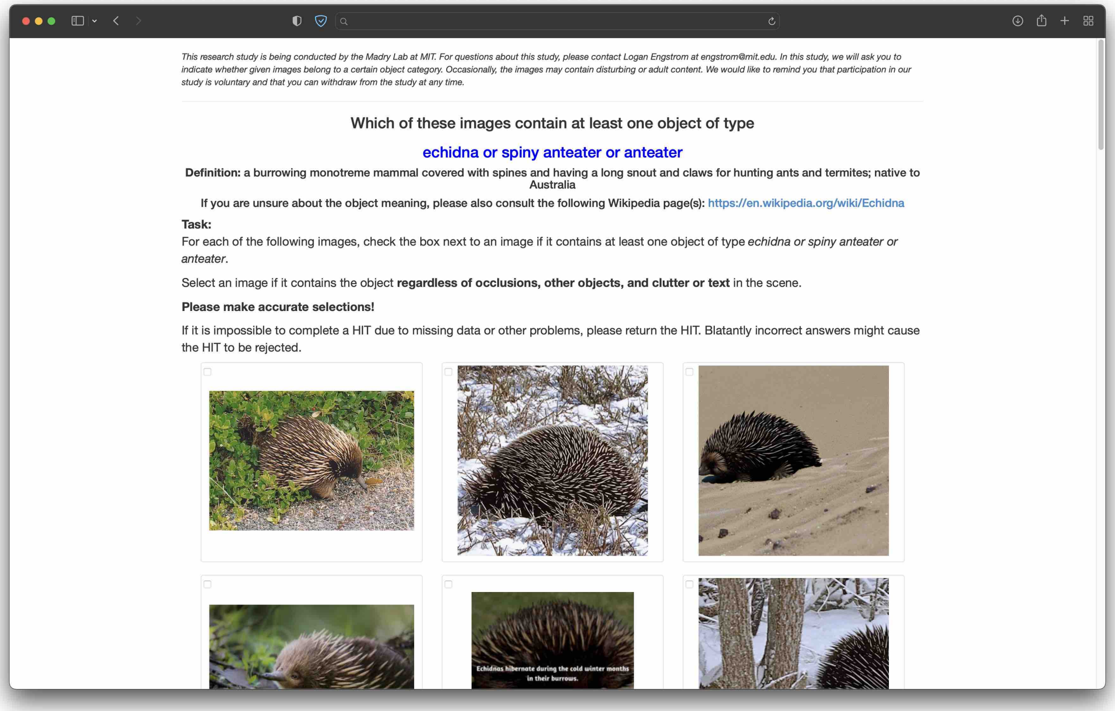
To set the similarity threshold for the presence of the object in a generated image, we evaluate the similarity between the embedding of c_{class} and every image of that class in the ImageNet validation set. We set then the threshold at the 20^{th} percentile of the CLIP similarities.

To set the similarity threshold for the presence of the distribution shift in a generated image, we first evaluate the CLIP similarity between the embedding of c_{shift} and every generated image in that distribution shift. For each of a fixed set of percentile values, we visually inspect a small number of images with similarities around that percentile, and select as our threshold the lowest percentile at which all inspected images exhibit the desired distribution shift.

A.4. User Study

We verify that our generated counterfactual examples for the ImageNet dataset contain the desired distribution shift and the object of interest through a user study on the Amazon Mechanical Turn (MTurk) crowd-sourcing platform. Below we describe the procedure of our study.

Procedure We send grids of 48 images to Amazon Mechanical Turk workers to label (pictured in Figure 10 and Figure 11). Each grid contains a single label, and the workers are asked to label all the instances of this label. This label depends on the task; for the ”label verification” task this label is an ImageNet label, and for the ”shift verification” this label is a distribution shift. We send each grid to 5 workers to label. We then measure the *selection frequency* for every image in the grid: the frequency at which workers selected the image as corresponding to the given label in the grid. We employ the selection frequency as a proxy score for how likely it is for the label to truly correspond to a given image.



This research study is being conducted by the Madry Lab at MIT. For questions about this study, please contact Logan Engstrom at engstrom@mit.edu. In this study, we will ask you to indicate whether given images belong to a certain object category. Occasionally, the images may contain disturbing or adult content. We would like to remind you that participation in our study is voluntary and that you can withdraw from the study at any time.

Which of these images contain at least one object of type

echidna or spiny anteater or anteater

Definition: a burrowing monotreme mammal covered with spines and having a long snout and claws for hunting ants and termites; native to Australia

If you are unsure about the object meaning, please also consult the following Wikipedia page(s): https://en.wikipedia.org/wiki/Echidna

Task:
For each of the following images, check the box next to an image if it contains at least one object of type *echidna or spiny anteater or anteater*.

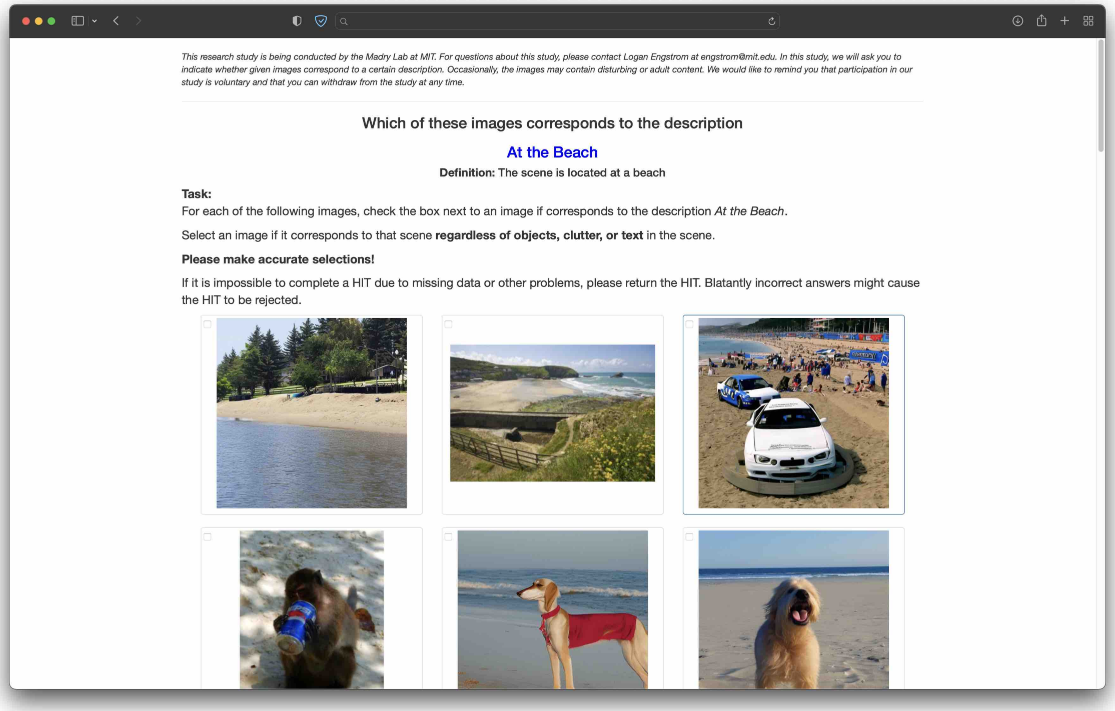
Select an image if it contains the object **regardless of occlusions, other objects, and clutter or text** in the scene.

Please make accurate selections!

If it is impossible to complete a HIT due to missing data or other problems, please return the HIT. Blatantly incorrect answers might cause the HIT to be rejected.

Echidnas hibernate during the cold winter months in their burrows.

Figure 10. User study: label task. In this task we ask crowd-workers to verify that the generated images correspond to the desired label.



This research study is being conducted by the Madry Lab at MIT. For questions about this study, please contact Logan Engstrom at engstrom@mit.edu. In this study, we will ask you to indicate whether given images correspond to a certain description. Occasionally, the images may contain disturbing or adult content. We would like to remind you that participation in our study is voluntary and that you can withdraw from the study at any time.

Which of these images corresponds to the description

At the Beach

Definition: The scene is located at a beach

Task:
For each of the following images, check the box next to an image if it corresponds to the description *At the Beach*.

Select an image if it corresponds to that scene **regardless of objects, clutter, or text** in the scene.

Please make accurate selections!

If it is impossible to complete a HIT due to missing data or other problems, please return the HIT. Blatantly incorrect answers might cause the HIT to be rejected.

Figure 11. User study: distribution shift task. In this task we ask crowd-workers to verify that the generated images correspond to the desired distribution shift.

A.5. Evaluating Distribution Shift Robustness

We generate 50000 images in 23 different shifts, listed in Table 1 along with the threshold we use to filter each shift. We evaluate the images on a sweep of image classification models obtained from the `timm` model repository (Wightman, 2019). We list all models in the file *models.txt* provided with the code.

Applying our CLIP metrics to filter the images results in an uneven number of images per class. In fact, for some of our more difficult shifts a class might have no images that pass the filter threshold. When calculating the accuracy of a model for images in a given shift, we only consider classes with at least five images remaining after filtering. Among these classes we measure the average per-class accuracy, and correspondingly measure the average per-class accuracy of the same classes among the base generated images. The difference between these two numbers is the accuracy drop we record for that model.

In Table 1 we list each of the distribution shifts we include in our benchmark along with the base prompt that we use to represent in-distribution samples. For each shift we list the full prompt that is input to Stable Diffusion, the CLIP threshold we set for filtering the generated images of that shift, and the % yield for images remaining after filtering with both metrics (for both presence of the desired distribution shift and object of interest).

Shift Name	Prompt	CLIP Threshold	% Yield
base	A photo of a <class>	–	92.6%
“in the grass”	A photo of a <class>in the grass	0.127	80.3%
“in the beach”	A photo of a <class>in the beach	0.175	62.9%
“in the forest”	A photo of a <class>in the forest	0.153	67.3%
“in the water”	A photo of a <class>in the water	0.163	60.1%
“on the road”	A photo of a <class>in the road	0.154	64.5%
“on the rocks”	A photo of a <class>in the rocks	0.124	76.6%
“in the snow”	A photo of a <class>in the snow	0.160	73.2%
“in the rain”	A photo of a <class>in the rain	0.173	48.3%
“in the fog”	A photo of a <class>in the fog	0.152	59.3%
“in bright sunlight”	A photo of a <class>in bright sunlight	0.124	89.9%
“at dusk”	A photo of a <class>at dusk	0.158	61.9%
“at night”	A photo of a <class>at night	0.147	61.1%
“studio lighting”	A photo of a <class>in studio lighting	0.140	66.6%
“blue”	A photo of a blue <class>	0.163	59.1%
“green”	A photo of a green <class>	0.190	51.3%
“red”	A photo of a red <class>	0.167	59.6%
“yellow”	A photo of a yellow <class>	0.212	43.3%
“orange”	A photo of a orange <class>	0.216	41.0%
“person and a”	A photo of a person and a <class>	0.181	29.9%
“and a flower”	A photo of a <class>and a flower	0.148	61.9%
“oil painting”	An oil panting of a <class>	0.214	67.2%
“pencil sketch”	A black and white pencil sketch of a <class>	0.223	61.8%
“embroidery”	An embroidery of a <class>	0.259	33.0%

Table 1. Full prompt, CLIP threshold for filtering, and % yield for each of the distribution shifts in our benchmark.

A.6. Public Release of the ImageNet* Benchmark

We publicly release the set of 1,000 learned tokens for ImageNet at ¡redacted¿, and we publicly release the our benchmark consisting of 23 different distribution shifts at ¡redacted¿.

B. Additional Results and Visualizations

Here we visualize additional experiments and examples extending upon the figures in the main paper. In Figure 12, we show visualize further examples of classes for which there is a visual mismatch between images generated by Stable Diffusion using the natural language prompts and images in ImageNet, as in Figure 12. In Figure 13, we show additional examples of using our dataset interface for model debugging, and as in Figure 6 we compare our counterfactual examples to those generated by prompting Stable Diffusion with natural language. In Figure 14, we take real images of plates with and without food and either in the grass or indoors to confirm the debugging results from Section 5. In Figure 15, we extend upon the visualizations in 7 and display additional samples of images from our distribution shift benchmark.

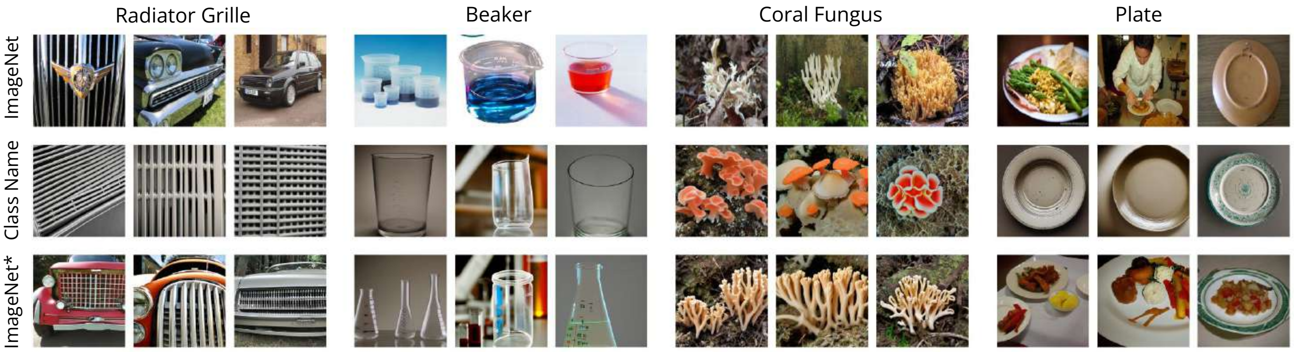


Figure 12. Additional examples of mismatch between prompting of Stable Diffusion using the class name and the ImageNet dataset. We visualize real images from ImageNet (**top**), images generated using the class name in prompts (**middle**) and ImageNet∗ (**bottom**).

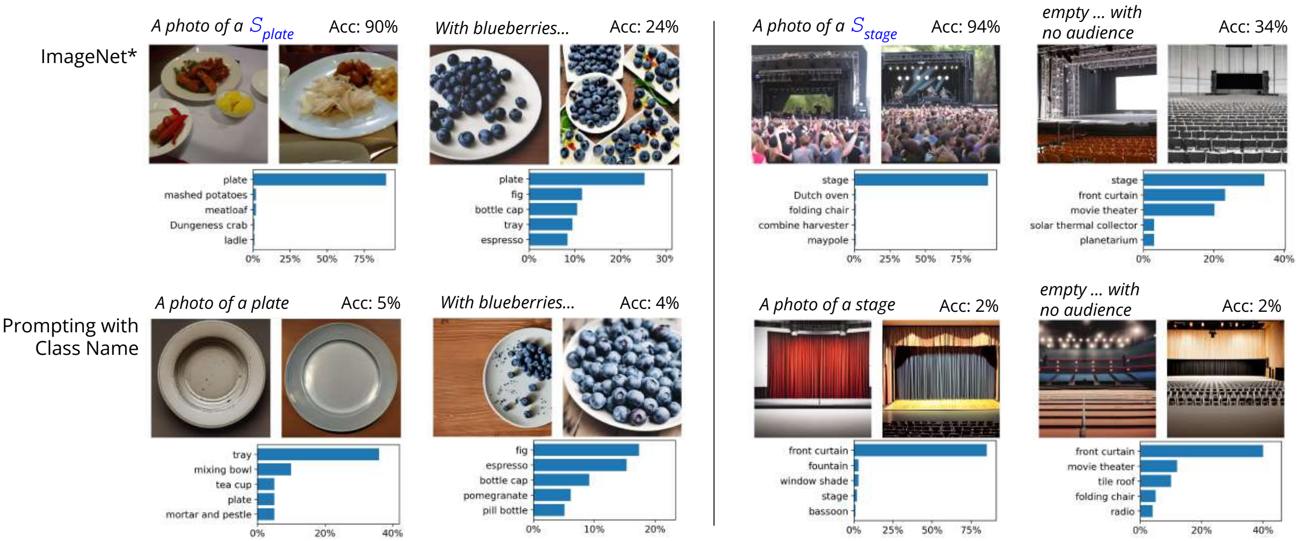


Figure 13. Additional images generated by our dataset interface ImageNet∗ (**top**) and the corresponding images generated by prompting Stable Diffusion with the class name (**bottom**), as well as the top predicted classes of an ImageNet-trained ResNet50. Using ImageNet∗, we find that “plate with blueberries” and “empty stage with no audience” both lead to a large degradation in the classifier’s accuracy compared to base generated images. On the other hand, the images generated by Stable Diffusion when prompted with the class name all lead to low model performance regardless of the specified shift.

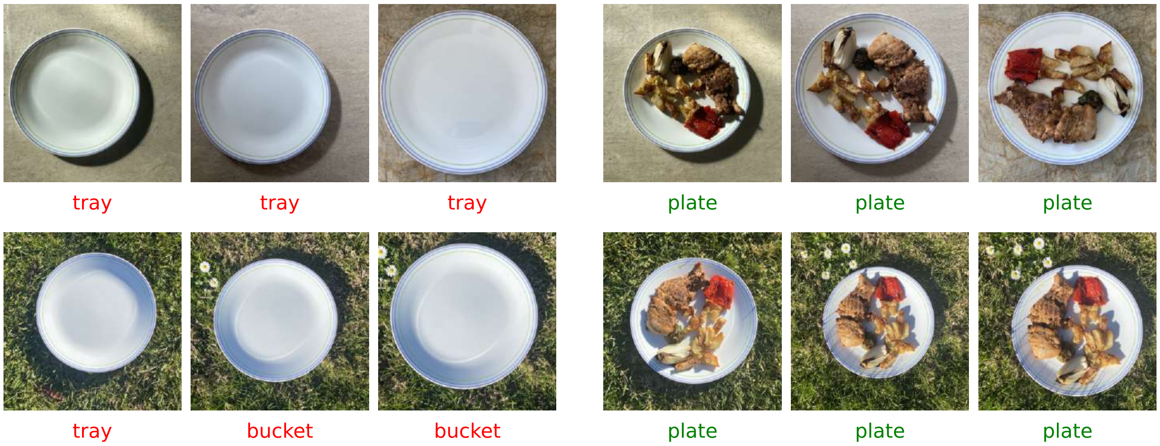


Figure 14. Real images of plates, with and without food and either on a table or in the grass. Below each image is the predicted class by an ImageNet-trained ResNet50.

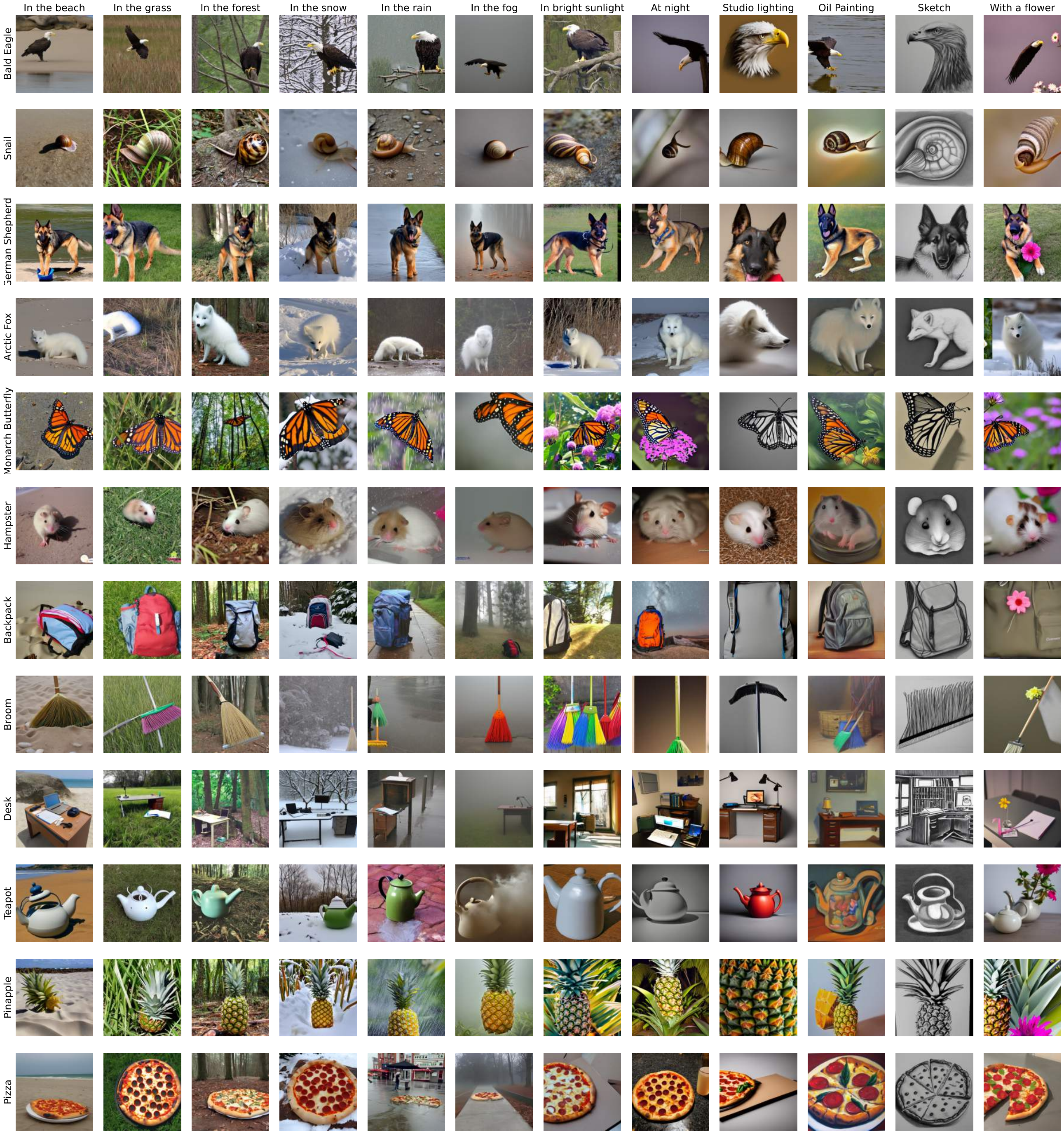
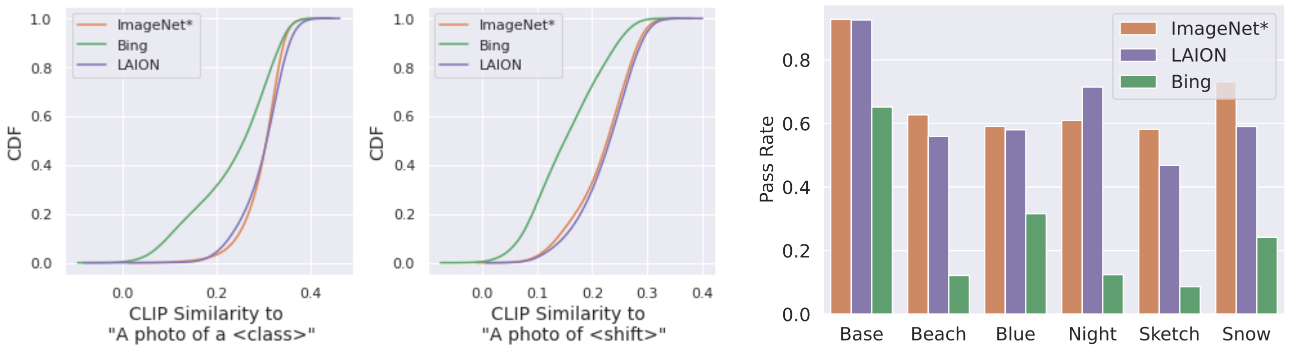


Figure 15. Additional examples of images generated with ImageNet∗ for a variety of distribution shifts. These images are a subset of the benchmark described in Section 6.



(a) CDF of CLIP similarity to captions c_{class} (**left**) and c_{shift} (**right**)

(b) Fraction of images passing the filters.

Figure 16. (**a**) CDFs of the CLIP similarity between the images and the captions c_{class} =*"A photo of a <class>"* and c_{shift} =*"A photo of a <shift>"*. (**b**) The fraction of images that pass the filters given the thresholds in Section TODO.

C. CLIP Metrics

In this section, we discuss how the filtering mechanisms discussed in Section 3.1 impact our generated dataset. Recall, that, for a given counterfactual example that is supposed to exhibit a class with a certain shift, we measure the CLIP similarity between the CLIP embedding of each generated image and the text embedding of the captions *"a photo of a <class>"* (c_{class}) and *"a photo <shift>"* (c_{shift}). We keep those that pass the threshold described in Appendix A.5.

In Figure 16a, we first plot the CDF of these CLIP scores for images scraped from Bing, ImageNet$*$, and images retrieved from via `clip-retrieval` with LAION for the shifts "at night", "blue", "in the beach","in the snow", and "sketch" as well as "base" images which do not correspond to a specific shift. We find that a higher proportion of ImageNet$*$ and LAION images have very high CLIP similarity to the corresponding captions than the Bing images. Note that `clip-retrieval` on LAION specifically takes images via a KNN on CLIP distance, and is thus pre-disposed to perform well on this metric.

In Figure 16b, we further calculate the proportion of images from each source that pass the automatic filters with the thresholds in Section A.5. In particular, ImageNet$*$ has the highest yield rate, with the majority of images passing the filters.

D. Experiments on Additional Datasets

In this section, we give qualitative results for applying our framework on two additional datasets, FGVC-Aircraft (Maji et al., 2013) and Oxford-IIIT Pet (Parkhi et al., 2012). For each of these two datasets, we learn a token for each class, as for ImageNet. Due to the fine-grained nature of the datasets and the possible confusion with specific labels (such as *E-170*, a type of aircraft), we initialize the Textual Inversion process for each class with the same broader description (*"pet"* and *"airplane"*).

In Figure 17 and 18, we visualize images generated using the learned tokens in a range of distribution shifts. Due to the more specific nature of the datasets, we are able to exhibit shifts that would not apply meaningfully across all of classes of a broader dataset such as ImageNet (e.g. *"through the clouds"* for FGVC-Aircraft, *"sleeping"* for Oxford-IIIT Pet).

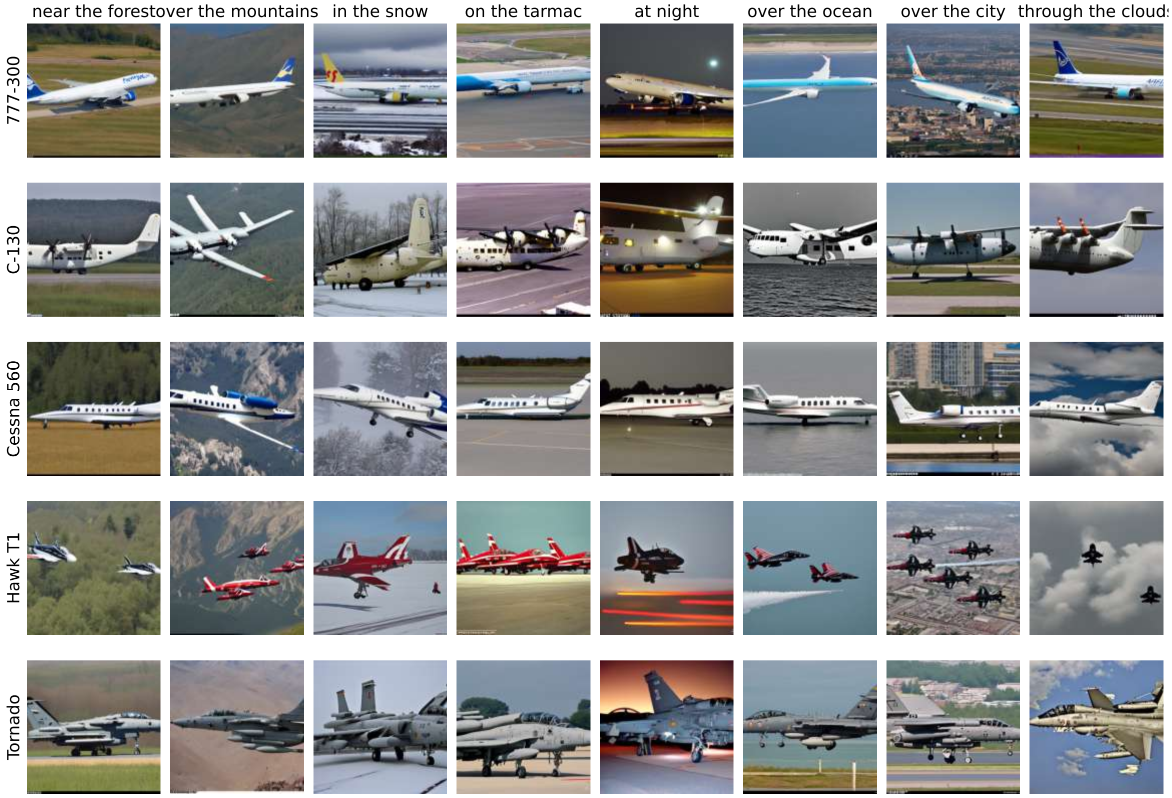


Figure 17. Examples of images generated with our learned tokens for the FGVC-Aircraft dataset in a variety of distribution shifts. These examples are not filtered with the CLIP similarity metrics.

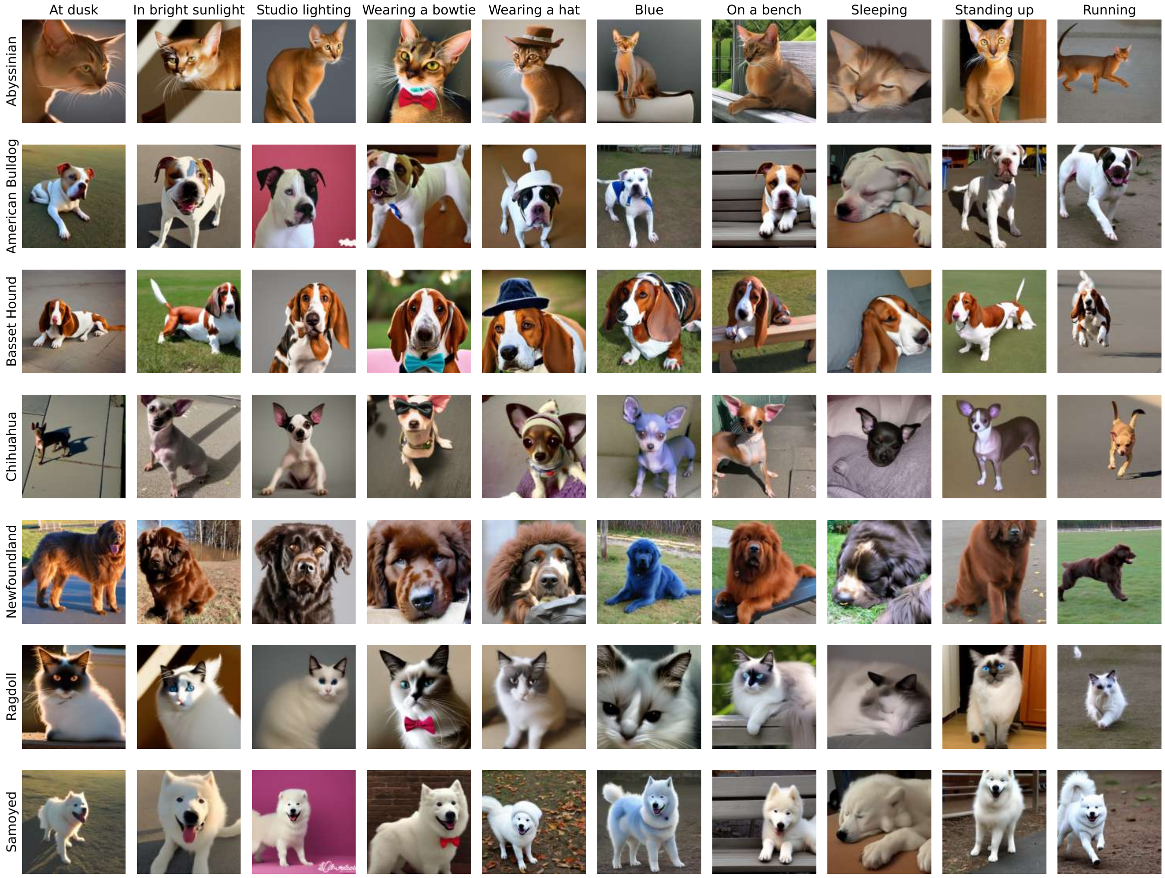


Figure 18. Examples of images generated with our learned tokens for the Oxford-IIIT Pet dataset in a variety of distribution shifts. These examples are not filtered with the CLIP similarity metrics.