

# Controlled Training Data Generation with Diffusion Models

Teresa Yeo\* Andrei Atanov\* Harold Benoit<sup>†</sup> Aleksandr Alekseev<sup>†</sup>  
 Ruchira Ray Pooya Esmaeil Akhoondi Amir Zamir

Swiss Federal Institute of Technology Lausanne (EPFL)

<https://adversarial-prompts.epfl.ch/>

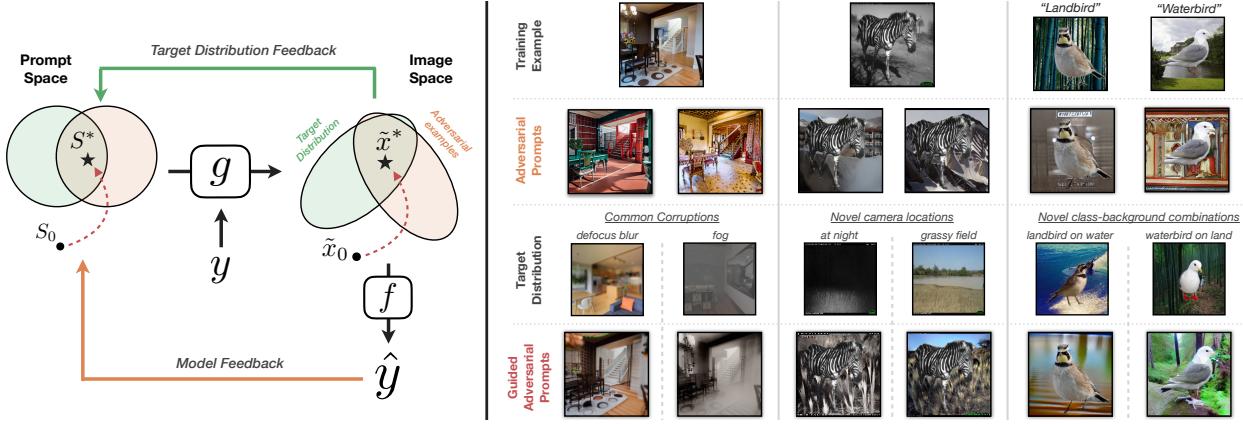


Figure 1. **A framework to generate model- and target distribution-informed training examples.** **Left:** An overview of how we generate training data for a given supervised model  $f$  and target distribution. Suppose  $g$  is a text-to-image generative model that generates images conditioned on a text prompt,  $S$  and label,  $y$ . A supervised model trained to perform a task, e.g., image classification or depth estimation, is denoted by  $f$ . We aim to find prompts that would generate training data useful for  $f$ , and we do so via two feedback mechanisms. The first mechanism makes use of feedback from  $f$ . In particular, we maximize the loss on the predictions from  $f$  to get **Adversarial Prompts**. The space of **Adversarial Prompts** and corresponding adversarial examples are represented by orange circles. Since there can be many adversarial examples not relevant to a particular target distribution, we introduce a second feedback mechanism that guides the prompts towards a certain target distribution (green circle). Combining the two mechanisms results in **Guided Adversarial Prompts**, represented by  $S^*$ . This results in generations that are both *relevant to the target distribution* and where  $f$  *does not perform well*. **Right:** We show some examples of generations attained by our method on several datasets and tasks. Namely, Taskonomy [80] (depth estimation), iWildCam [3] (animal classification), and Waterbirds [66] (bird classification). An exemplar training image is shown in the first row and the third row illustrates examples of target distributions. The second row shows the generations from **Adversarial Prompts** which uses only model feedback. It finds novel “styles” that fool a given model  $f$ , but may not match a target distribution (third row). **Guided Adversarial Prompts** (fourth row) uses feedback from the target distribution and generates images that both fool the model and are similar to the target.

## Abstract

In this work, we present a method to control a text-to-image generative model to produce training data specifically “useful” for supervised learning. Unlike previous works that employ an open-loop approach and pre-define prompts to generate new data using either a language model or human expertise, we develop an automated **closed-loop** system which involves **two feedback mechanisms**. The first mechanism uses feedback from a given supervised model and finds **adversarial** prompts that result in image generations that maximize the model loss. While these adversarial prompts result in diverse data informed by the model, they are not informed of the target distribution, which can be inefficient. Therefore, we introduce the second

feedback mechanism that **guides** the generation process towards a certain target distribution. We call the method combining these two mechanisms **Guided Adversarial Prompts**. We perform our evaluations on different tasks, datasets and architectures, with different types of distribution shifts (spuriously correlated data, unseen domains) and demonstrate the efficiency of the proposed feedback mechanisms compared to open-loop approaches.

## 1. Introduction

The quality of data plays a crucial role in training generalizable deep learning models [21, 52, 74]. For a model to generalize well, its training data should be representative of the test distribution where it will be deployed. However,

real world test conditions change over time, while training datasets are typically collected once and remain static due to high collection costs. We, therefore, focus on generating datasets that can adapt to novel test distributions and are more cost-efficient.

Diffusion generative models [33, 55, 61, 67, 70] are trained on large-scale collections of images [69] and exhibit remarkable generalization abilities by being able to produce realistic images not seen during training. Additionally, unlike static datasets that they are trained on, these generative models allow us to *adapt* the generation process to produce images that follow a certain conditioning. For example, they can be conditioned on textual prompts [61] or geometric information such as depth maps [82].

Recent works explore the use of diffusion models to generate training data for supervised learning with promising results [18, 28, 68]. They guide the generation process using text prompts to accomplish two goals: produce aligned image-label pairs for supervised training and adapt the generated images to a certain target distribution. These proposed methods, however, find conditioning text prompts in an open-loop way by either using a language model [18] or heuristics [68]. Therefore, they *lack an automatic feedback mechanism* that can refine the found text prompts to produce more curated and useful training data. Furthermore, it has been argued that being able to control the input data is a key contributor to how children are able to learn with few examples [6, 44].

In this work, we propose two feedback mechanisms to find prompts for generating useful training data. The first mechanism finds prompts that result in generations that maximize the loss of a particular supervised model, thus, *reflecting its failure modes*. We call them Adversarial Prompts (AP). This mechanism ensures that we find not only novel prompts, which may produce images that the model already performs well on, but adversarial prompts that produce images with high loss, and, thus, useful for improving the model [14] (see exemplar generations for AP in Figs. 1, 3 and 5)

A given model can perform poorly on multiple distribution shifts, and traversing all of them with adversarial optimization to adapt it to a specific target distribution can be inefficient (e.g., see the difference between AP generations and the illustrated target distributions in Fig. 1-right). Therefore, we introduce an additional *target-informed* feedback mechanism that finds prompts that generate images similar to those from the target distribution we want to adapt to. To implement this, we assume either access to a textual description of the target distribution or a few sampled (unlabeled) images from it. We then optimize a similarity metric between CLIP [57] embeddings of the generated examples and the target description. We call these prompts Guided Adversarial Prompts (GAP). Compare the columns Adver-

sarial Prompts and Guided Adversarial Prompts in Figs. 1 and 3 to 5 to see the effect of CLIP guidance in steering the generations towards a specific target distribution.

We demonstrate the effectiveness of our method on different tasks (image classification, depth estimation), datasets with distribution shifts (Waterbirds [66], iWildCam [4, 41], Common Corruptions [29], 3D Common Corruptions [37]), and architectures (convolutional and transformer) with supportive results.

## 2. Related Work

**Open-loop data generation** methods use pre-defined controls to guide the generative process and produce novel training examples. One line of work uses GANs [8, 35, 59] and pre-define a perturbation in their latent space to generate novel examples. More recent works adopt text-to-image diffusion models and use pre-defined prompt templates [27, 68, 77] or use a language model to generative variations of a given prompt [77]. These methods require *anticipating the kind of data that will be seen at test-time* when defining the prompts. On the other hand, our CLIP guidance mechanism allows us to generate images similar to the target distribution. [18] also approach this problem by using a captioning and language model to summarize a target distribution shift into a text prompt. However, this summarization process is not informed of the generations, and, thus, does not guarantee that the text prompt will guide the generation process to images related to the target distribution. Finally, these methods are not model-informed and do not necessarily generate images *useful* for training a given model.

**Closed-loop data generation** methods guide the generation process via an automatic feedback mechanism. They control the latent space of GANs [5] or VAEs [76] models, NeRF [15], or the space of hand-crafted augmentations [10] to generate data that maximizes the loss of the network on the generated data. Similarly, [36] uses an SVM to identify the failure modes of a given model and uses this information to generate training data with a diffusion model. Our method employs a similar adversarial formulation (in conjunction with target distribution guidance) but performs the optimization in the text prompt space of recently developed diffusion models.

**“Shallow” data augmentation** techniques apply simple hand-crafted transformations to training images to increase data diversity and improve the model’s generalization. Examples of such transformations are color jitter, random crop, and flipping, etc. To produce more diverse augmentations, methods like RandAugment [11] and AugMix [30] combine multiple of such simple transformations, and Mixup [81] and CutMix [78] methods use transformations that can combine multiple images. AutoAugment [10] and adversarial training [49] build a closed system to tune

the parameters of the applied augmentations but are inherently limited by the expressiveness of the simple transformations. In contrast, our method uses expressive diffusion models, which results in images that are more diverse and realistic than those produced by “shallow” augmentations.

**Controlling diffusion models.** Methods like ControlNet [82] and T2I-Adapter [54] adapt a pre-trained diffusion model to allow for additional conditioning e.g., edge, segmentation, and depth maps. We employ these models for generation as it allows us to generate paired data for different tasks, given the labels from an existing dataset. Editing methods aim to modify a given image, either via the prompt [32], masks [9], instructions [7] or inversion of the latent space [34, 53]. In contrast, personalization methods aim to adapt diffusion models to a given concept e.g., an object, individual, or style. Popular examples include textual inversion [22] and DreamBooth [64], which aim to find a token to represent a concept given several images of that concept. The former freezes the diffusion model, while the latter fine-tunes it. Extensions of these works learn to represent multiple concepts [1, 24]. In our work, we adopt an approach similar to textual inversion to steer the diffusion model, but our method can also be used with other controlling mechanisms.

### 3. Method

We begin this section by formalizing our problem setting and describing how diffusion models can be used to generate training data (Sec. 3.1). We then introduce two feedback mechanisms to find prompts that are informed of the failure modes of a given model (Sec. 3.2) and relevant to a given target distribution (Sec. 3.3).

#### 3.1. Preliminaries

**Problem Formulation.** We consider the problem of supervised learning, where a model  $f : \mathcal{X} \rightarrow \mathcal{Y}$  learns a mapping from the image space  $\mathcal{X}$ , to a target space  $\mathcal{Y}$ , e.g., a depth estimation or semantic classification problem. The model  $f$  is trained using training dataset  $\mathcal{D}_{\text{train}}$  and tested on a new set  $\mathcal{D}_{\text{test}}$  that exhibits a distribution shift w.r.t. the training data. Our goal is to generate additional *synthetic* training data  $\mathcal{D}_{\text{syn}}$  to adapt the model and improve its performance under the distribution shift. To apply the target-informed feedback mechanism described in Sec. 3.3, we assume access to some information about the test distribution, either from text descriptions or a few samples of *unlabeled* images.

**Text-to-image Diffusion Models.** We use the Stable Diffusion [62] text-to-image diffusion model as the basis for our generator  $g$ . Given a textual prompt  $c$ , Stable Diffusion is capable of synthesizing realistic images following the textual conditioning. However, in general, for a given task, e.g., depth estimation, a textual prompt alone may not

be sufficient for controlling the generation well enough to produce aligned image-label examples.

**Generating aligned training examples.** We employ the following two approaches to condition the generative model  $g$  on the label  $y$  and sample aligned training examples  $(\tilde{x}, y)$ . For the depth estimation task, we use the ControlNet [83] model which extends the conditioning mechanisms of the Stable Diffusion to accept various spatial modalities, e.g., depth maps, segmentation masks, or edges. Specifically, we use ControlNet v1.0 with depth conditioning<sup>1</sup>. For semantic classification tasks, we utilize the foreground object masks and use an in-painting technique proposed in [47] that preserves the masked region throughout the denoising process, essentially keeping it intact. These mechanisms provide us with a generative model conditioned both on a text prompt  $c$  and label  $y$ . We denote the resulting distribution modeled by this generative model as  $g(y, c)$ .

#### 3.2. Model-Informed Generation with Adversarial Prompt Optimization

Our first feedback mechanism aims at generating training examples that reflect the failure modes of a given model  $f$ . An automatic way to do so is via **adversarial optimization**, which finds the “worst case” failure modes of  $f$ . More precisely, we find a text prompt  $c$  that generates images  $\tilde{x} \sim g(y, c)$  that maximize the supervised loss  $\mathcal{L}(f(\tilde{x}), y)$ , e.g.,  $l_1$  loss for depth estimation. Since the usual prompt space is discrete (text tokens) and challenging to optimize over, we employ the approach introduced in Textual Inversion [22] and instead optimize over the corresponding continuous embedding space. For ease of notation, “prompt space” will implicitly refer to the continuous embedding space instead of the discrete token space. We construct a prompt  $c_w$  out of  $n$  new “placeholder” tokens, i.e.,  $c_w = (c_{w_1}, \dots, c_{w_n})$ , and find their corresponding embedding weights  $\{w_i\}_{i=1}^n$  by solving the following optimization problem:

$$w_{\text{AP}} = \arg \min_w \mathbb{E}_y \mathbb{E}_{\tilde{x} \sim g(y, c_w)} \mathcal{L}_{\text{adv}}(f(\tilde{x}), y), \quad (1)$$

where  $\mathcal{L}_{\text{adv}} = -\mathcal{L}$  and  $y$  is sampled from  $\mathcal{D}_{\text{train}}$ . Note that the sample  $\tilde{x}$  is differentiable w.r.t. the embeddings  $w$  which allows us to use gradient-based optimization. We call the prompts that result from solving the above optimization problem Adversarial Prompts (AP).

**Avoiding  $(\tilde{x}, y)$  alignment collapse.** The adversarial objective in Eq. (1) aims to fool the model  $f$ . However, it may instead fool the label-conditioning mechanism of the generative model  $g$ , resulting in  $c_{w_{\text{adv}}}$  generating samples  $\tilde{x} \sim g(y, c_{w_{\text{adv}}})$  that are not faithful to  $y$  (see Fig. 2). To avoid this, we further constrain the expressiveness of the generation process. There are several ways to do so.

<sup>1</sup><https://github.com/lillyasviel/ControlNet>



Figure 2. **Ways to alleviate the misalignment of the generation with its conditioning.** The third column onwards shows some examples of generations from depth maps that *do not follow* the depth conditioning. See the first and second columns for the original image and its depth label. There are several ways to constrain the generation to alleviate this misalignment. **1.** Early stopping involves stopping the adversarial optimization when the loss reaches a certain threshold. The resulting generations from early stopping are shown in the fourth column. **2.** SDEdit [51] involves conditioning the generation process on the original image. This mechanism is applied during generation with the adversarial prompts i.e., applying SDEdit to the prompts that generated the images in the third column results in the last column generations. Both SDEdit and early stopping are able to improve the alignment of the generations with depth conditioning.

One way is to use the SDEdit method [50], which conditions the generation process on the original image by starting the denoising process from a noised version of  $x$  instead of pure noise. Thus, it constrains the expressive power of the generative model to produce samples closer to the original image  $x$ .

Additionally, some constraints can be implemented w.r.t  $\mathcal{L}_{\text{adv}}$ . For the depth estimation task, we employ an early stopping criterion and stop the adversarial optimization when the loss reaches a certain task-specific threshold. For semantic classification, choosing  $\mathcal{L}_{\text{adv}}$  to be the negative cross-entropy loss, although natural, may not be a good choice. Indeed, for iWildCam, although we keep the class mask intact, we observed that optimizing the negative cross-entropy loss may lead to the generation of another class somewhere else in the image, e.g. an elephant is generated next to a giraffe, destroying the  $(\tilde{x}, y)$  alignment. Thus, for iWildCam, we choose to maximize the uncertainty or entropy of the model’s prediction on the generated images. We provide more details in the Appendix Sec. 7.5.2.

Finally, our CLIP [57] guidance loss introduced in Sec. 3.3 further constrains possible perturbations to a target distribution and helps to avoid the generation of non-realistic images.

### 3.3. Target Distribution Informed Generation

The adversarial formulation above finds prompts that reflect the failure modes of  $f$ . Without any information about the target distribution, improving the model on the worst-performing distributions is one of the best strategies one can do and, indeed, improves performance in some cases (see Fig. 4 and Fig. 6a). However, there are typically multiple failure modes of a given model and many possible dis-

tribution shifts that can occur at test-time. Adapting to all of them using only the first feedback mechanism could be inefficient when the goal is to adapt to a specific target distribution instead of improving the performance on average. Thus, we introduce the second feedback mechanism to inform the prompt optimization process of the target distribution. This only requires access to simple text descriptions (e.g., ‘fog’ to adapt to foggy images) or a small number ( $\sim 100$ ) of **unlabelled** images.

We implement the target-informed feedback mechanism using CLIP [57] guidance. Specifically, we assume access to either textual descriptions of the target image distribution  $\{t_j\}$ , a few unlabeled image samples  $\{x_j\}$  or both. We then construct the corresponding text and image guidance embeddings as  $e_t = \text{avg}(\{E_t(t_j)\})$  and  $e_i = \text{avg}(\{E_i(x_j)\})$ , where  $E_t$  and  $E_i$  denote, respectively, the CLIP text and image encoders, and avg stand for averaging. We then use the following guidance loss:

$$\mathcal{L}_{\text{CLIP}}(\tilde{x}, c_w) = \lambda_t \mathcal{L}_t(E_t(c_w), e_t) + \lambda_i \mathcal{L}_i(E_i(\tilde{x}), e_i), \quad (2)$$

where we take  $\mathcal{L}_t$  to be  $l_2$  norm between two embeddings and  $\mathcal{L}_i$  to be the negative cosine similarity, as we found it to perform the best. See the Appendix Sec. 7.7 for the results of this ablation. Note, that based on the available information, one can also use only one of the two guidance losses. Finally, we combine both adversarial, Eq. (1), and CLIP guidance, Eq. (2), losses to form the final objective:

$$w_{\text{GAP}} = \arg \min_w \mathbb{E}_y \mathbb{E}_{\tilde{x} \sim g(y, c_w)} [\mathcal{L}_{\text{adv}}(f(\tilde{x}), y) + \mathcal{L}_{\text{CLIP}}(\tilde{x}, c_w)] \quad (3)$$

We call the prompts that result from solving Eq. (3), Guided Adversarial Prompts (GAP). See the Appendix Secs. 7.4.2 and 7.5.3 for further implementation details.

## 4. Experiments

We perform experiments in three settings: domain generalization via camera trap animal classification on the iWildCam [4] dataset, bird classification with spurious correlation with the Waterbirds [66] dataset, and depth estimation with the Taskonomy dataset [79, 80]. For depth estimation, the considered distribution shifts are Common Corruptions [29] (CC), 3D Common Corruptions [37] (3DCC) applied on the Taskonomy [80] test set and cross dataset shift from the Replica [73] dataset.

### 4.1. Semantic Classification

**Waterbirds** [66] is a dataset constructed by pasting an image of either a waterbird or landbird from the CUB [75] dataset, which represents the label  $y$ , onto a “land” or “water” background image from the Places [84] dataset. We follow [18] and take only images of waterbirds appearing



**Figure 3. On Waterbirds, Guided Adversarial Prompts are able to generate counterfactual examples not present in the original training dataset and improves the data-efficiency over other methods.** **Left:** We train a classification model on the original spuriously correlated dataset (see examples on the right) with varying number of extra data points generated using different types of prompts. We measure the accuracy on a balanced set where waterbirds and landbirds appear on both land and water. We run each experiment with three seeds and report the mean and standard deviation. **Right:** We show examples of generated images with each type of prompts on the Waterbirds. We also provide original training examples where the background is a perfectly predictive feature of the bird type. A red frame signifies that the model trained only on the original data (“No Extra Data”) misclassifies the image, and a green frame stands for the correct prediction. We observe that having a guidance mechanism towards the target image distribution consistently improves on top of the **Agnostic Prompts** baseline (“nature” prompt). **Adversarial Prompts**, while fooling the model, generates images that are different from the target distribution and, thus, not useful to adapt the model to it. Combining both mechanisms in **Guided Adversarial Prompts** leads to **improved data efficiency**. Unlike **Guided Prompts** that uses the same prompts to generate images for both classes, **GAP** finds prompts that generate images the model fails on, this leads to generation of waterbirds on land and landbirds on water, the combinations not present in the original training data, which are necessary data samples for the model to learn the bird predictive feature.

on water and landbirds on land background as  $\mathcal{D}_{\text{train}}$ , i.e., in the training data, the background alone is predictive of the bird class. The examples from the test distribution  $\mathcal{D}_{\text{test}}$  contain all four combinations of bird class and background.

**iWildCam** [3, 42] is a domain generalization dataset, made up of a large-scale collection of images captured from camera traps placed in various locations around the world. We seek to learn a model that generalizes to photos taken from new camera deployments. We follow [18] and subsample the dataset to create a 7-way classification task (background, cattle, elephant, impala, zebra, giraffe, dik-dik), with 2 test locations that are not in the training or validation set. We also fixed the number of additional generated images for finetuning to 2224 images in order to match the setting of [18].

For each dataset, we compare the following methods (we provide more details in Appendix Sec. 7):

**No Extra Data:** We train a ResNet50 [26] model using the original training data  $\mathcal{D}_{\text{train}}$  without using any extra data.

**Augmentation baselines:** We compare to two data augmentation baselines taken from recent literature: CutMix [78] and RandAugment [12].

**Agnostic Prompts:** We use a prompt that is not informed of the model or the target distribution. Similar to ALIA [18], we use a prompt “*nature*” for Waterbirds and the prompt template “*a camera trap photo of {class name}*”.

**Guided Prompts (ALIA) [18]:** This approach uses a captioning and language model to summarize a target distribution shift into text prompts. Specifically, we use the prompts found by the ALIA method. This results in seven prompts for Waterbirds and four prompts for iWildCam. See Appendix Sec. 7.2 for more details and a discussion

on the differences between ALIA and our method.

**Adversarial Prompts:** We use the model trained without extra data as the target model  $f$  and find adversarial prompts following Eq. (1). We find four prompts per class for Waterbirds, eight in total, and four prompts in total applied to all classes for iWildCam.

**Guided Adversarial Prompts:** We use the same setting as in Adversarial Prompts and apply additional CLIP guidance to adapt to a target distribution shift, following Eq. (3). For Waterbirds, we apply text guidance using ALIA prompts as the textual description of the target distribution. For iWildCam, we use image guidance and partition the target test distribution into four groups based on two attributes that have significant impact on the visual characteristics of the data; the test location (first or second) and time of the day (day or night). We sample 64 *unlabelled* images randomly from each group. We optimize for one guided adversarial prompt per group.

**Training details.** For both datasets, we perform adversarial optimization with constant learning rate of  $1 \times 10^{-3}$  using Adam [38]. We use the DDIM [72] scheduler. See Tab. 1 for a summary of the experimental parameters. The adversarial loss  $\mathcal{L}_{\text{adv}}$  from Eq. (1) is defined as the negative cross-entropy loss for Waterbirds. As mentioned in Sec. 3.2, we optimize the entropy loss for iWildCam. More precisely, this loss is equal to the cross entropy loss where the target label  $y$  is replaced by the soft label  $\tilde{y} = \{\frac{1}{|\mathcal{Y}|}, \dots, \frac{1}{|\mathcal{Y}|}\}$ , the uniform distribution over all classes. This loss explicitly encourages generations that either (1) do not contain new animals (2) contain new animals that are not accounted for in the label space  $\mathcal{Y}$ . For more details, see the Appendix.

**Waterbirds results.** Fig. 3-left shows the performance

Table 1. Adversarial optimization parameters for the classification experiments.

Dataset	SDEdit strength	denoising steps	guidance scale	# placeholder tokens	Opt. steps (AP/GAP)	$\lambda_t/\lambda_i$
Waterbirds	1.	5	7.0	5	1000/1000	20/0
iWildCam	0.8	5	5.0	10	2000/10000	0/10

of each method, and Fig. 3-right demonstrates the corresponding generated examples for each method. First, we find that while Adversarial Prompts works on par with the Agnostic Prompts in a low-data regime, it performs worse with more generated data. Looking at the corresponding generations, we see that, while being adversarial, these images are different from the target distribution, which can explain the inferior performance on this particular target distribution. Second, using text prompts informed of the target leads to a consistent improvement over the Agnostic Prompts, and the corresponding images look more similar to the real images from the target distribution. Finally, Guided Adversarial Prompts combining both feedback mechanisms results in more data-efficient generations outperforming all other methods in the low-data regime. Looking at the corresponding generations, we can see that GAP tends to generate only useful combinations, i.e., waterbirds on land and landbirds on water that are missing in the training dataset. This can explain its higher sample efficiency.

Table 2. **Model feedback generates data tailored to a specific model.** We show test accuracy (averaged over 3 seeds) on iWildCam for different combinations of model and generated data. Each row represents the model that was used for model-based feedback to get Guided Adversarial Prompts. Each column represents the model used for finetuning on the generated data. *The best dataset for each model is the one generated using that model.*

Data	Finetuned model	
	ResNet	ViT
Agnostic Prompts	72.94	73.07
GAP from ResNet	<b>83.97</b>	72.61
GAP from ViT	83.87	<b>77.21</b>

**iWildCam results.** Fig. 4 shows that having model-informed feedback (Adversarial Prompts) helps to generate more useful data than no feedback mechanism (Agnostic Prompts) and also improves the performance of the target-only informed Guided Prompts method in the low-data regime. Guided Adversarial Prompts combines the benefits of both model- and target-informed feedback mechanisms, consistently outperforming other methods. Fig. 4 shows exemplar generations for each method. We find that while AP generates images distinct from the target distribution and images generated by target-informed methods (snow background vs. grass background), training a model using these examples in the low-data regime performs better than GP and similar to GAP.

**Fine-tuning with Guided Adversarial Prompts from a different model.** To evaluate our method’s ability to customize data to a model,  $f$ , and assess its resilience to changes in model architecture, we changed the model from a ResNet50 [26] to a ViT-B-16 [16] model. In Tab. 2, we investigate whether prompts optimized using one model feedback can be useful for finetuning another model, and conclude that model feedback does indeed generate data more useful for that particular model. Furthermore, the data generated from one model when used to fine-tune another model can still significantly improve performance over Agnostic Prompts. See Appendix Fig. 14 for the results of Fig. 4 with a ViT-B-16 model, similar trends hold.

## 4.2. Depth Estimation

For depth estimation, we consider the following pre-trained models as  $f$ : **1**) a U-Net [63] model trained on the Taskonomy dataset [79, 80] and **2**) a dense prediction transformer (DPT) [58] model trained on Omnidata [20].

We compare the following methods. They all involve fine-tuning  $f$ , but on different datasets. We use ControlNet v1.0 with depth conditioning for the experiments in this section. See Fig. 5 for a comparison of the generations:

**Control (No extra data):** We fine-tune  $f$  on the original training data. This baseline is to ensure that the difference in performance is due to the generated data, rather than e.g., longer training or optimization hyperparameters.

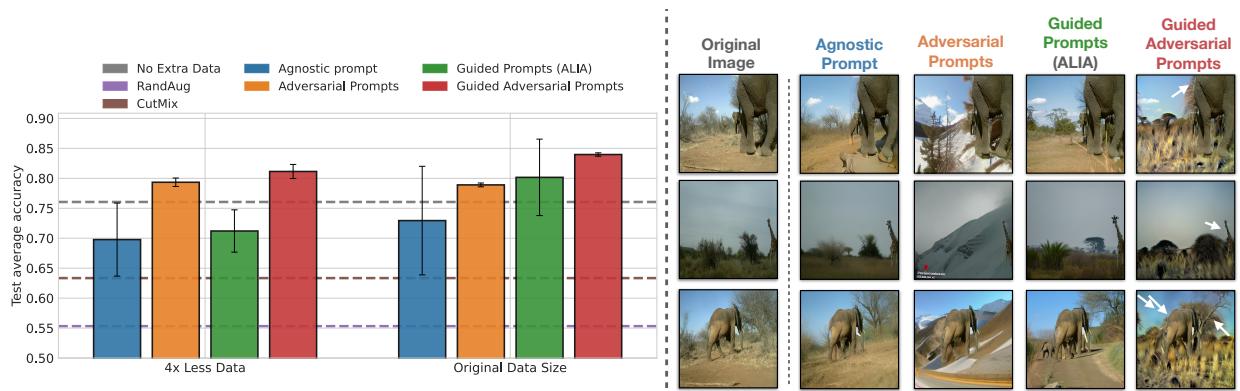
**Agnostic Prompts:** This baseline generates data that is *agnostic* to the model or the target distribution. We generate images with the prompt “room” as the datasets consist of indoor images from mostly residential buildings.

**Agnostic Prompts (Random):** We generate data with “random” prompts. In our proposed method, we optimize for  $n$  embedding vectors, resulting in a prompt,  $c$ . Thus, to match this setting, from a Gaussian distribution fitted on the embeddings from the vocabulary, we sample  $n$  random embeddings to create a random prompt to be used in the data generation.

**Adversarial Prompts:** We perform the optimization as described in Eq. (1) and fine-tune on this data.

**Guided Prompts:** We perform the optimization using only the loss described in Eq. (2) and fine-tune on this data.

**Guided Adversarial Prompts:** In addition to optimizing the adversarial loss, we also optimize the CLIP guidance loss as described in Eq. (3). This allows us to generate data that



**Figure 4. Left: For iWildCam, Guided Adversarial Prompts are superior in performance and data-efficiency.** We train a model on the combination of the original training data and extra data generated using different types of prompts. We show the average accuracy on two iWildCam test camera trap locations. We run each experiment with three seeds and report the mean and standard deviation. **(1) Guided Adversarial Prompts** consistently outperforms **Guided Prompts** and **Adversarial Prompts**. For **AP** and **GAP**, there is a much lower performance drop (1-3%) when reducing the amount of generated data compared to **GP** (9%). This suggests that being only target-informed requires a more exhaustive exploration of the image space to find samples that are “useful” for the pre-trained model, compared to being model-informed. **Right: Qualitative results and comparison with ALIA on the iWildCam dataset.** From left to right: the real training data from iWildCam; generation using the **Agnostic Prompts** template: “*a camera trap photo of {class name}*”; **Guided Prompts (ALIA)** generations with target-informed prompt template: “*a photo of {class name} in a grassy field with trees and bushes*”; **Adversarial Prompts** generations; and **Guided Adversarial Prompts** generations with the same target distribution as **Guided Prompts**. **GAP** generations, while being visually coherent with the “grassy field” location, introduce an adversarial “camouflage” effect. White arrows point to the animals for clarity. From top to bottom, they are (mis)classified as cattle, dik-dik, cattle, cattle.

is also informed of a certain distribution shift.

**Training details.** The adversarial optimization was done with AdamW [46], learning rate of  $5.0 \times 10^{-4}$ , weight decay of  $1.0 \times 10^{-3}$ , and batch size of 8. We set the early stopping threshold (mentioned in Sec. 3.2) to 0.08 for the UNet model and 1.0 for the DPT model. They were trained with  $\ell_1$  and Midas loss [20] respectively. We perform a total of 30 runs to get different Adversarial Prompts. We use the DDIM [72] scheduler. During optimization, we use only 5 denoising steps, as it is more stable. For Guided Adversarial Optimization, the guidance coefficient for text and image guidance is 1 and 5 respectively. For fine-tuning, we generate images with 15 steps. For the GP runs with SDEdit, we used strength 0.6, for the GAP runs, strength 0.9. See the Appendix Sec. 8.1 for further details.

**Comparing the generated images with different prompts.** Fig. 5-left shows the results of the generations for the baselines and our method, optimized on the Taskonomy dataset. The generation with Agnostic Prompts are visually different from that of the original image, however, they tend to have similar styles. In contrast, the generations with Adversarial Prompts have more complex styles and are more diverse. Using SDEdit during the optimization and generation results in generations that are closer to the original image, as it was also used as conditioning. The last four columns show the results of using CLIP text guidance for the target distribution shift *fog* and *blur*, as described in Sec. 3.3, with and without adversarial optimization. The generations with Guided Prompts involve passing the depth conditioning and prompt “fog” or “blur” to the diffusion model. In both cases, the generations result in

a mild level of fog or blur. In contrast, Guided Adversarial Prompts results in more severe fog and blur corruptions. See Appendix Sec. 8.4.2 for generations using image guidance.

Note that all the generations follow the conditioning, i.e., depth labels (see first column). See Sec. 3.2 for the discussion on how we prevent the generations from collapsing. Thus, this gives us *aligned training data*, i.e., RGB images and depth labels that we can use for fine-tuning.

**Performance on OOD data.** We evaluate our method and the baselines after fine-tuning on their respective generated datasets. Tab. 6a demonstrated that data generated with Adversarial Prompts improve the performance of the model under different distribution shifts. Furthermore, we show that the trend holds with the DPT model. Thus, our method is able to successfully find useful Adversarial Prompts for different architectures.

**Performance of GAP against amount of generated data.** Fig. 5-right shows the performance of our method on the *focus blur* corruption over the amount of extra generated data (See Appendix Fig. 18 for results on other corruptions). Guided Prompts or Guided Adversarial Prompts results in a large improvement in performance, compared to Adversarial Prompts or the baseline with only 10 extra data points. This suggests that the guidance loss *successfully steered the generations toward producing training data relevant to the distribution shift*. This experiment was performed with image guidance. In Appendix Sec. 8.4.2, we compare the qualitative and quantitative differences from image and text guidance. Text guided prompts tends to generate corruptions that are more realistic

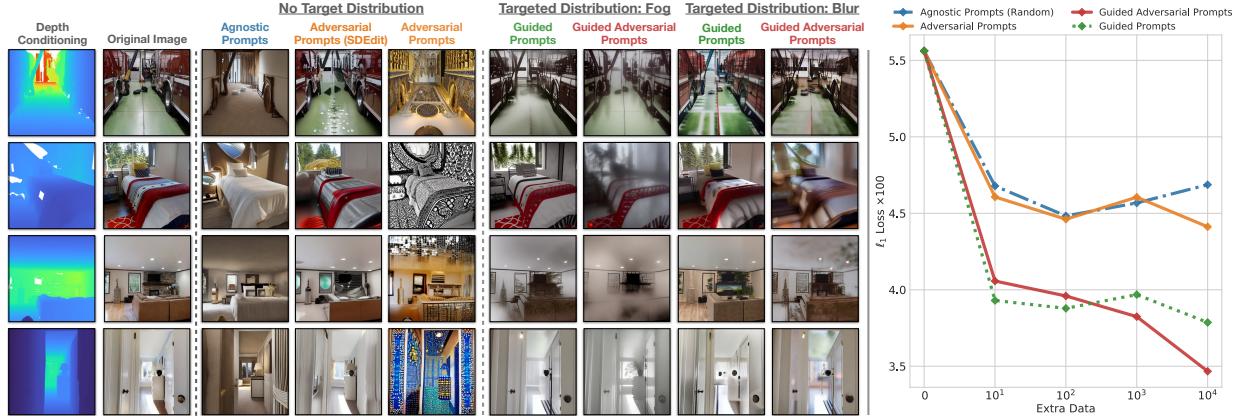


Figure 5. **Left: A comparison of generations with different prompts.** A comparison of the generated data for the baselines and our method. Generations with **Adversarial Prompts** results in *diverse styles* that are *distinct from the original training data* (see fifth column). **Adversarial Prompts** with SDEdit results in generations shown in the fourth column. As these generations are conditioned on the original image, they look more similar to them. The last 4 columns show the generations with text guidance for target shifts *fog* and *blur*. Using **Guided Prompts** alone, in this case “*fog*” or “*blur*”, results in generations with a mild fog or blur. Performing **Guided Adversarial Prompts** results in generations with more severe fog or blur. **Right: Performance of GAP with different amount of added data.** The distribution shift, in this case, is *defocus blur* applied on the Taskonomy test set. The plot shows the  $\ell_1$  loss ( $\times 100$ ) of the U-Net model versus the amount of extra data generated and used for fine-tuning. In this example, we performed image guidance with a set of 100 *unlabelled* corrupted RGB images. The plot shows that both **GP** and **GAP** were able to guide the optimization toward generating training data relevant to the distribution shift. Furthermore, there is a large improvement compared to using **AP** or the baselines with as little as 10 extra generated samples. See Appendix Fig. 18 for results on other corruptions.

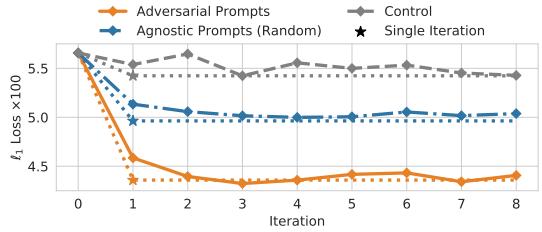
(a) **Quantitative results on depth estimation.**  $\ell_1$  errors on the depth prediction task for a pre-trained U-Net and DPT model. (Lower is better. U-Net losses are multiplied by 100 and DPT losses by 10, for readability). Note that these models were trained with different losses,  $\ell_1$  for the former and Midas loss [19] for the latter, thus their performance is not comparable. We evaluate on distribution shifts from Common Corruptions (CC), 3D Common Corruptions (3DCC), and cross-datasets (CDS), Replica. The results from CC and 3DCC are averaged over all distortions and severity levels on Taskonomy. Our method is able to generate training data that can improve results over the baselines on several distribution shifts. Performing **AP** with SDEdit gives better results than **AP** under distribution shifts. Thus, conditioning on the original image seems to be helpful for these shifts. For the DPT model, the trends are similar, **AP** performs better than the baselines. See Appendix Tab. 6 for results on other baselines, in particular, data augmentation ones.

	U-Net				DPT	
	Taskonomy			Replica	Taskonomy	
Shift	Clean	CC	3DCC	CDS	CC	3DCC
Control (No extra data)	<b>2.35</b>	4.93	4.79	5.38	3.76	3.42
<b>Agnostic Prompts</b>	2.47	5.03	4.17	5.30	4.06	3.58
<b>Agnostic Prompts (Random)</b>	2.38	4.96	4.11	5.14	3.88	3.51
<b>Adversarial Prompts</b>	2.49	4.36	4.02	5.12	3.40	3.28
<b>Adversarial Prompts (SDEdit)</b>	2.59	<b>4.20</b>	<b>3.88</b>	<b>4.96</b>	<b>3.35</b>	<b>3.25</b>

Figure 6. **Left:** Results for Adversarial Prompts against the baselines. **Right:** Results from performing multiple iterations of adversarial optimization and finetuning.

(see Appendix Fig. 18 for comparisons). However, image guidance tends to perform better quantitatively across the different target distributions.

Unlike classification, the performance of Guided Adversarial Prompts for depth is not consistent across all distribution shifts. The diffusion model was not able to generate certain shifts e.g., noise corruptions, as the text descriptions and unlabelled images was too ambiguous e.g., ‘noise’ or having common attributes other than the corruption. We leave further analysis to future work.



(b) **Comparing the performance from running multiple iterations versus a single iteration** of adversarial optimization, generation, and fine-tuning. The plot shows the  $\ell_1$  loss ( $\times 100$ ) of the U-Net model against the number of iterations on the depth prediction task. The  $\ell_1$  loss is computed on the Taskonomy dataset under common corruptions, averaged over all corruptions and severity levels. The single- and multi-iteration runs have similar settings in terms of the total number of **AP** used, the same number of tokens per prompt, etc. See Appendix Sec. 8.4.1 for further details. The first iteration of the multi-iteration run resulted in the largest improvement in performance. However, it converges to a similar performance as the single iteration run. Thus, we chose to perform only a single iteration in Fig. 6a.

**Running multiple iterations of adversarial optimization vs a single iteration.** We define an *iteration* as one round of adversarial optimization, i.e. optimizing Eq. (1) or Eq. (3), generation and fine-tuning. Given that all of the above results were obtained with a single iteration, we aim to see if there are benefits in performing multiple iterations. We perform a total of 8 iterations and we compare this to performing a single iteration. The experimental settings for 8 iterations and a single iteration are similar, e.g., in total, over the 8 iterations, we optimize for the same number of

Adversarial Prompts, and fine-tune on the same datapoints, etc. Fig. 6b shows that the first iteration of the 8 iterations setting resulted in the largest improvement in performance, eventually converging to the performance of the single iteration approach. Thus, we chose to perform a single iteration for the results in Fig. 6a. See the Appendix Sec. 8.4.1 for further implementation details and results.

## 5. Conclusion and Limitations

In this work, we aim to generate training data useful for training a supervised model by steering a text-to-image generative model. We introduced two feedback mechanisms to find prompts that are informed by both the given model and the target distribution. Evaluations on a diverse set of tasks and distribution shifts show the effectiveness of the proposed closed-loop approach in comparison to open-loop ones. Below we briefly discuss some of the limitations:

*Label shift:* In this work, we focus on generating novel images. However, some distribution shifts can also change the label distribution, e.g., for depth estimation, changing from indoor to outdoor scenes would result in a shift in depth maps. One possible approach could be learning a generative model over the label space [43] to control the generation in both the label and image space.

*Computational cost:* Estimating the gradient of the loss in Eq. (3) requires backpropagation through the denoising process of the diffusion model, which can be computationally demanding. Using approaches that reduce the number of denoising steps [48, 71] may be able to reduce this computational cost.

*Label Conditioning:* As discussed in Sec. 3.2, our method is limited by the faithfulness of the generation conditioned on the given label. For example, we found that the semantic segmentation ControlNet does not follow the conditioning accurately enough to be useful for the supervised model. Further developments in more robust conditioning mechanisms are needed to successfully apply our method to other tasks.

## References

- [1] Omri Avrahami, Kfir Aberman, Ohad Fried, Daniel Cohen-Or, and Dani Lischinski. Break-a-scene: Extracting multiple concepts from a single image. *arXiv preprint arXiv:2305.16311*, 2023. 3
- [2] Sara Beery, Dan Morris, and Siyu Yang. Efficient Pipeline for Camera Trap Image Review, 2019. *arXiv:1907.06772 [cs]*. 15
- [3] Sara Beery, Arushi Agarwal, Elijah Cole, and Vighnesh Birodkar. The iwildcam 2021 competition dataset. *arXiv preprint arXiv:2105.03494*, 2021. 1, 5, 14, 15
- [4] Sara Beery, Arushi Agarwal, Elijah Cole, and Vighnesh Birodkar. The iwildcam 2021 competition dataset. *arXiv preprint arXiv:2105.03494*, 2021. 2, 4
- [5] Victor Besnier, Himalaya Jain, Andrei Bursuc, Matthieu Cord, and Patrick Pérez. This dataset does not exist: training models from generated images. In *ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 1–5. IEEE, 2020. 2
- [6] Oliver Braddick and Janette Atkinson. Visual control of manual actions: brain mechanisms in typical development and developmental disorders. *Developmental Medicine and Child Neurology*, 55 Suppl 4:13–18, 2013. 2
- [7] 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. 3, 13
- [8] Lucy Chai, Jun-Yan Zhu, Eli Shechtman, Phillip Isola, and Richard Zhang. Ensembling with Deep Generative Views, 2021. *arXiv:2104.14551 [cs]*. 2
- [9] Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. Diffedit: Diffusion-based semantic image editing with mask guidance. *arXiv preprint arXiv:2210.11427*, 2022. 3
- [10] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018. 2
- [11] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020. 2
- [12] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020. 5
- [13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 14
- [14] Zhun Deng, Linjun Zhang, Kailas Vodrahalli, Kenji Kawaguchi, and James Zou. Adversarial Training Helps Transfer Learning via Better Representations, 2021. *arXiv:2106.10189 [cs]*. 2
- [15] Yinpeng Dong, Shouwei Ruan, Hang Su, Caixin Kang, Xingxing Wei, and Jun Zhu. Viewfool: Evaluating the robustness of visual recognition to adversarial viewpoints. *Advances in Neural Information Processing Systems*, 35: 36789–36803, 2022. 2
- [16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, and others. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 6, 18
- [17] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, and others. An image is worth 16x16 words:

- Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 13
- [18] Lisa Dunlap, Alyssa Umino, Han Zhang, Jiezhi Yang, Joseph E Gonzalez, and Trevor Darrell. Diversify your vision datasets with automatic diffusion-based augmentation. *arXiv preprint arXiv:2305.16289*, 2023. 2, 4, 5, 13, 14, 15, 20
- [19] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3D scans. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10786–10796, 2021. 8
- [20] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A Scalable Pipeline for Making Multi-Task Mid-Level Vision Datasets from 3D Scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10786–10796, 2021. 6, 7, 20
- [21] Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, Eyal Orgad, Rahim Entezari, Giannis Daras, Sarah Pratt, Vivek Ramanujan, Yonatan Bitton, Kalyani Marathe, Stephen Mussmann, Richard Vencu, Mehdi Cherti, Ranjay Krishna, Pang Wei Koh, Olga Saukh, Alexander Ratner, Shuran Song, Hannaneh Hajishirzi, Ali Farhadi, Romain Beaumont, Sewoong Oh, Alex Dimakis, Jenia Jitsev, Yair Carmon, Vaishaal Shankar, and Ludwig Schmidt. DataComp: In search of the next generation of multimodal datasets, 2023. *arXiv:2304.14108 [cs]*. 1
- [22] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022. 3, 13, 18
- [23] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018. 20
- [24] Ligong Han, Yinxiao Li, Han Zhang, Peyman Milanfar, Dimitris Metaxas, and Feng Yang. Svdiff: Compact parameter space for diffusion fine-tuning. *arXiv preprint arXiv:2303.11305*, 2023. 3
- [25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 14
- [26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 5, 6, 13
- [27] Ruifei He, Shuyang Sun, Xin Yu, Chuhui Xue, Wenqing Zhang, Philip Torr, Song Bai, and Xiaojuan Qi. Is synthetic data from generative models ready for image recognition? *arXiv preprint arXiv:2210.07574*, 2022. 2
- [28] Ruifei He, Shuyang Sun, Xin Yu, Chuhui Xue, Wenqing Zhang, Philip Torr, Song Bai, and Xiaojuan Qi. Is synthetic data from generative models ready for image recognition? *arXiv preprint arXiv:2210.07574*, 2022. 2
- [29] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019. 2, 4, 21
- [30] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019. 2
- [31] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, and others. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8340–8349, 2021. 20
- [32] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022. 3
- [33] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2
- [34] Inbar Huberman-Spiegelglas, Vladimir Kulikov, and Tomer Michaeli. An edit friendly DDPM noise space: Inversion and manipulations. *arXiv preprint arXiv:2304.06140*, 2023. 3
- [35] Ali Jahanian, Xavier Puig, Yonglong Tian, and Phillip Isola. Generative models as a data source for multiview representation learning. *arXiv preprint arXiv:2106.05258*, 2021. 2
- [36] Saachi Jain, Hannah Lawrence, Ankur Moitra, and Aleksander Madry. Distilling model failures as directions in latent space. *arXiv preprint arXiv:2206.14754*, 2022. 2
- [37] Oğuzhan Fatih Kar, Teresa Yeo, and Amir Zamir. 3D Common Corruptions for Object Recognition. In *ICML 2022 Shift Happens Workshop*, 2022. 2, 4, 21
- [38] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, 2017. *arXiv:1412.6980 [cs]*. 5, 15
- [39] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes, 2022. *arXiv:1312.6114 [cs, stat]*. 18, 20
- [40] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment Anything, 2023. *arXiv:2304.02643 [cs]*. 14
- [41] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, and others. Wilds: A benchmark of in-the-wild distribution shifts. In *International conference on machine learning*, pages 5637–5664. PMLR, 2021. 2
- [42] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, and others. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021. 5
- [43] Guillaume Le Moing, Tuan-Hung Vu, Himalaya Jain, Patrick Pérez, and Matthieu Cord. Semantic palette: Guiding

- scene generation with class proportions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9342–9350, 2021. 9
- [44] Terri L. Lewis and Daphne Maurer. Multiple sensitive periods in human visual development: evidence from visually deprived children. *Developmental Psychobiology*, 46(3):163–183, 2005. 2
- [45] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation, 2022. arXiv:2201.12086 [cs]. 13
- [46] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization, 2019. arXiv:1711.05101 [cs, math]. 7, 20
- [47] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. RePaint: Inpainting using Denoising Diffusion Probabilistic Models, 2022. arXiv:2201.09865 [cs]. 3, 13
- [48] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent Consistency Models: Synthesizing High-Resolution Images with Few-Step Inference, 2023. arXiv:2310.04378 [cs]. 9
- [49] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 2
- [50] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jianjun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. 4, 13, 16
- [51] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jianjun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. 4
- [52] 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. 1
- [53] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6038–6047, 2023. 3
- [54] Chong Mou, Xintao Wang, Liangbin Xie, Jian Zhang, Zhonggang Qi, Ying Shan, and Xiaohu Qie. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. *arXiv preprint arXiv:2302.08453*, 2023. 3
- [55] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models, 2022. arXiv:2112.10741 [cs]. 2
- [56] OpenAI. GPT-4 Technical Report, 2023. arXiv:2303.08774 [cs]. 13
- [57] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and others. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 2, 4
- [58] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12179–12188, 2021. 6, 13
- [59] Suman Ravuri and Oriol Vinyals. Classification Accuracy Score for Conditional Generative Models, 2019. arXiv:1905.10887 [cs, stat]. 2
- [60] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019. 20
- [61] 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. 2, 13, 18
- [62] 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. 3
- [63] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International conference on medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 6, 13
- [64] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22500–22510, 2023. 3
- [65] 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. *arXiv preprint arXiv:1911.08731*, 2019. 14
- [66] 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. *arXiv preprint arXiv:1911.08731*, 2019. 1, 2, 4, 13
- [67] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamvar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding, 2022. arXiv:2205.11487 [cs]. 2
- [68] Mert Bulent Sarıyıldız, Kartek Alahari, Diane Larlus, and Yannis Kalantidis. Fake it till you make it: Learning transferable representations from synthetic ImageNet clones. In *CVPR 2023—IEEE/CVF conference on computer vision and pattern recognition*, 2023. 2

- [69] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5B: An open large-scale dataset for training next generation image-text models, 2022. arXiv:2210.08402 [cs]. [2](#)
- [70] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015. [2](#)
- [71] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. [9](#), [14](#)
- [72] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. [5](#), [7](#), [20](#)
- [73] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica Dataset: A Digital Replica of Indoor Spaces. *arXiv preprint arXiv:1906.05797*, 2019. [4](#), [21](#)
- [74] Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. *arXiv preprint arXiv:2007.00644*, 2020. [1](#)
- [75] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. *The Caltech-UCSD Birds-200-2011 Dataset*. California Institute of Technology, 2011. [4](#)
- [76] Eric Wong and J Zico Kolter. Learning perturbation sets for robust machine learning. *arXiv preprint arXiv:2007.08450*, 2020. [2](#)
- [77] Jianhao Yuan, Francesco Pinto, Adam Davies, Aarushi Gupta, and Philip Torr. Not just pretty pictures: Text-to-image generators enable interpretable interventions for robust representations. *arXiv preprint arXiv:2212.11237*, 2022. [2](#)
- [78] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019. [2](#), [5](#)
- [79] Amir Zamir, Alexander Sax, Teresa Yeo, Oğuzhan Kar, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas Guibas. Robust Learning Through Cross-Task Consistency. *arXiv preprint arXiv:2006.04096*, 2020. [4](#), [6](#), [20](#)
- [80] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3712–3722, 2018. [1](#), [4](#), [6](#)
- [81] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. [2](#)
- [82] Lvmin Zhang and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *arXiv preprint arXiv:2302.05543*, 2023. [2](#), [3](#)
- [83] Lvmin Zhang and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *arXiv preprint arXiv:2302.05543*, 2023. [3](#)
- [84] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning Deep Features for Scene Recognition using Places Database. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2014. [4](#)

# Controlled Training Data Generation with Diffusion Models

## Appendix

### 6. Outline

We provide further discussions, details, and evaluations in the appendix, as outlined below.

- Secs. 7 and 8 describe additional implementation details for our classification and depth estimation experiments, respectively.
- Sec. 7.6 describes an image guidance mechanism using Textual Inversion [22] and compares it with the CLIP guidance mechanism, on Waterbirds.
- Sec. 8.2 provide additional results for “standard” augmentation baselines for the depth estimation experiments.
- Secs. 7.4.3, 7.5.4 and 8.3 provide **qualitative generations from all the Adversarial Prompts and Guided Adversarial Prompts** used in the Waterbirds, iWildCam, and depth estimation experiments.
  - Additionally, for depth estimation, we provide a qualitative comparison of Adversarial Prompts generations optimized on different models (UNet [63], DPT [58]). For iWildCam, we also provide additional results using a ViT-B-16 [17] instead of a ResNet50 [26].
- Sec. 8.4 provides additional analysis on the depth estimation experiments:
  - the single iteration vs. multi-iteration setting
  - a comparison of CLIP image and text guidance
  - **an assessment of the generalization of Adversarial Prompts from one model to another.**

### 7. Classification

#### 7.1. Training data generation

**Inpainting.** As mentioned in main paper Sec. 3.1, for semantic classification tasks, we utilize the foreground object masks and use an in-painting technique proposed in [47] that preserves the masked region throughout the denoising process. In this section, we briefly describe this procedure and refer the reader to the original work for more details.

Let  $m$  be a binary pixel mask, where a pixel is equal to 1 if the pixel contains the object and 0 otherwise, and  $x$  be the original image from a training dataset. During generation, after obtaining a denoised sample  $\tilde{x}_t$  at time  $t$  we update it as  $\tilde{x}_t \leftarrow m \odot x_t^{\text{orig}} + (1-m) \odot \tilde{x}_t$ , where  $x_t^{\text{orig}}$  is the original image noised to have the correct properties of the expected Gaussian distribution at time  $t$ .

However, because we are using Stable Diffusion [61], the denoising process is done in latent space (using an encoder  $\mathcal{E}$ ), not pixel space. This means that to apply inpainting, we must resize the mask  $m$  to the latent space dimensions, and apply the above-described procedure in the la-

tent space:  $\tilde{z}_t \leftarrow m_z \odot z_t^{\text{orig}} + (1 - m_z) \odot \tilde{z}_t$ , where  $z_0^{\text{orig}} = \mathcal{E}(x^{\text{orig}})$  and  $z_t^{\text{orig}}$  is its corresponding noised version. While this procedure usually performs well in preserving the original region of interest, we also paste the original masked region in the pixel space to obtain the final sample  $\tilde{x} = m \odot x^{\text{orig}} + (1 - m) \odot \tilde{x}_0$ .

**SDEdit [50].** In addition to inpainting, depending on the setting, we also use SDEdit [50], a mechanism available to all diffusion models that allows to use an initial image to condition the generation of new images to be closer to the initial image. The mechanism is parametrized by the *SDEdit strength*  $s$ , which indicates the extent by which the model can deviate from the original image.

**Text-to-image model.** For our diffusion model, we use Stable Diffusion v1.5<sup>2</sup>.

#### 7.2. ALIA

Here, we give more details on the ALIA [18] (Automated Language-guided Image Augmentation) baseline method, which aims at generating images targeting a particular test distribution similar to our guidance mechanism (main paper Sec. 3.3).

Given exemplar images from the test distribution, ALIA first captions each image using the BLIP [45] captioning model. Then, it uses the GPT-4 [56] LLM to summarize these captions into a list of domains asking it to produce descriptions that are agnostic to the class information. [18] then use these prompts to generate additional training data. In order to preserve the original class information in their generations, they use SDEdit [50] or Instruct Pix2Pix [7]. We refer the original paper for further implementation details. Below, we summarize resulting prompts we use for comparison in our results.

For Waterbirds [66], we found that removing the prefix “*a photo of a {class name}*” from the original prompts when using the inpainting technique (Sec. 7.1) to work slightly better for both the ALIA baseline and our CLIP text guidance (main paper Eq. (2)). We, therefore, use the following prompts:

- “*in a bamboo forest with a green background.*”
- “*flying over the water with a city skyline in the background.*”
- “*perched on a car window.*”
- “*standing in the snow in a forest.*”
- “*standing on a tree stump in the woods.*”

<sup>2</sup><https://huggingface.co/runwayml/stable-diffusion-v1-5>

Dataset	ALIA			Ours		
	SDEdit strength	sampling steps	text guidance	SDEdit strength	sampling steps	text guidance
Waterbirds	0.3	50	7.0	1.	15	7.0
iWildCam	0.5	50	7.5	0.8	5	5.0

Table 3. Generation parameters for the classification experiments

Dataset	Training		
	learning rate	weight decay	epochs
Waterbirds	0.001	1e-4	100
iWildCam	0.0001	1e-4	100/20

Table 4. Training parameters for the classification experiments

- “swimming in a lake with mountains in the background.”,
  - “standing on the beach looking up.”
- For iWildCam [3], we keep the original prompts intact:
- “a camera trap photo of a {class name} in a grassy field with trees and bushes.”
  - “a camera trap photo of a {class name} in a forest in the dark.”
  - “a camera trap photo of a {class name} near a large body of water in the middle of a field.”
  - “a camera trap photo of a {class name} walking on a dirt trail with twigs and branches.”

There are two main differences between ALIA and our method:

- The target distribution feedback.** ALIA aligns its prompts with the target distribution by utilizing captioning and summarizing. However, this summarizing process is not informed of the produced generations when using such prompts, and, thus, does not guarantee that the text prompt will accurately guide the generation process to images related to the target distribution.
- Model feedback.** ALIA is not model-informed. Thus, it doesn’t necessarily generate images *useful* for training a given model.

Those two differences originate from the fact that ALIA is an **open-loop** method, i.e., it lacks the mechanism to refine the prompt based on the generated images. In contrast, our method uses model and target distribution feedback in a **closed-loop**. This allows our method to outperform ALIA and be more data-efficient.

### 7.3. General implementation details

**Generation.** We report our generation parameters in Tab. 3. We use the DDIM [71] scheduler. We generate 384x384 resolution images. Those parameters were chosen based on visual inspection, ease of optimization and downstream performance (validation accuracy). **Training data.** After generation, ALIA’s method consists of an additional filtering step to remove “bad” generations. This step relies

on using a pretrained model to measure confidence on the generated images. However, given our method creates images that are adversarial to an iWildCam pretrained model, the filtering part of ALIA’s pipeline is not usable on our data. Thus, to keep things comparable, we decided not to apply filtering both our method generated data and ALIA’s generated data. However, it must be noted that [18] only reports a 2% absolute accuracy drop between filtering and no filtering on iWildCam (1.4% on Waterbirds), thus we do not expect a big difference in performance with ALIA’s reported results and our results.

**Supervised Training.** We report our training parameters in Tab. 4. We use ALIA’s codebase to finetune our models, which ensures fair comparison to the ALIA baselines. For everything except the generated data, the settings are the same as in ALIA. For both datasets, the starting model is a ResNet50 [25] model, pretrained on ImageNet [13].

The reported test accuracy is chosen according to the best checkpoint, measured by validation accuracy.

## 7.4. Waterbirds

### 7.4.1 Dataset details

Fig. 7 demonstrates the shift between train and test distributions in the Waterbirds dataset [65]. We follow the setting suggested in [18] and use 1139 images as  $\mathcal{D}_{\text{tr}}$ , where waterbirds appear only on water background and landbirds on land background. We add additional 839 examples either from the original dataset, where waterbirds appear only on land background and landbirds on water background (“Real OOD data”), or generated by Stable Diffusion with prompts obtained by one of the methods. For the data-efficiency plots (e.g., Fig. 3) we reduced the number of added examples by a factor of  $\{1/2, 1/4, 1/8, 1/16\}$ .

Since the original Waterbirds dataset does not provide masks for the exact generated images, we used the SAM [40] segmentation model to obtain bird segmentation masks for training images. We use these masks to condition the generative model on the class by using inpainting as described in Sec. 7.1.

### 7.4.2 Implementation Details

**Adversarial Optimization.** For adversarial feedback, we use the model trained only using the original training data  $\mathcal{D}_{\text{train}}$  with complete spurious correlation. It is taken from ALIA checkpoints<sup>3</sup>. As the task is the binary classification, we use the cross-entropy loss for the opposite class as the

<sup>3</sup><https://api.wandb.ai/files/clipinvariance/ALIA-Waterbirds/y6zc932x/checkpoint/ckpt-Waterbirds-none-filtered-resnet50-1-0.001-0.0001/best.pth>

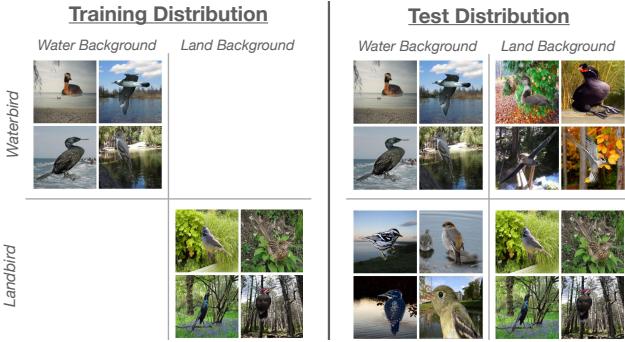


Figure 7. **Distribution shift in the Waterbirds dataset.** The background is a perfectly predictive spurious feature on the training distribution, but loses predictive power in the test distribution.

adversarial loss:  $\mathcal{L}_{\text{adv}}(\tilde{x}, y) = \mathcal{L}_{\text{x-ent}}(f(\tilde{x}), 1 - y)$ , assuming  $y \in \{0, 1\}$ . This is equivalent to the negative cross-entropy loss referred in the text. We find four prompts per each class, i.e., eight prompts in total. Each prompt is composed of five new learnable tokens. We perform adversarial optimization for 1000 steps with learning rate 1e-3 using Adam [38]. We use five denoising steps during adversarial optimization and generate images for training with 15 steps. We do not use SDEdit for Waterbirds. See Tab. 3 for summary.

**CLIP Guidance.** For Waterbirds, we use CLIP text guidance by encoding each of ALIA’s summarized prompts (see Sec. 7.2) with the CLIP text encoder as described in main paper Sec. 3.3. In addition, we renormalize the averaged target text embedding to have the norm equal to the mean norm of the original prompts, and use the resulting vector as the target  $e_t$ . We use  $l_2$  guidance loss:  $\mathcal{L}_t(E_t(c_w), e_t) = \|E_t(c_w) - e_t\|_2^2$ . We use  $\lambda_t = 20$  and  $\lambda_i = 0$  (i.e., no image guidance).

#### 7.4.3 Additional Qualitative Results.

In Fig. 8 and Fig. 9, we show a few generations using all 8 prompts used in the Waterbirds experiments for Guided Adversarial Prompts and Adversarial Prompts, respectively.

### 7.5. iWildCam

#### 7.5.1 Dataset details.

The original iWildCam [3] dataset is subsampled to create a 7-way classification task (background, cattle, elephant, impala, zebra, giraffe, dik-dik). The training set has 6,000 images with some classes having as few as 50 images per example. There are 2 test locations that are not in the training or validation set. Additionally, given  $h$ , the hour at which an image was taken, we define an image to be during “daytime” if  $9 \leq h \leq 17$ , and

“nighttime” if  $h \leq 5 \vee h \geq 20$ . As said in main paper Sec. 4.1, for image CLIP guidance, we separate the target test locations into four groups ( $\text{location}=\{1, 2\}$ ,  $\text{time}=\{\text{daytime}, \text{nighttime}\}$ ). We provide visualisation of the test locations (at day & night) in Fig. 10. For more details on the iWildCam subset construction, we refer to [18] Section 8.3. For inpainting, the object masks are obtained from MegaDetector [2].

#### 7.5.2 Alignment collapse solution for iWildCam.

As mentioned in main paper Sec. 3.1, choosing  $\mathcal{L}_{\text{adv}}$  to be the negative cross entropy loss, i.e. minimizing the probability that the model predicts  $y$ , may not be the best choice. Indeed, given we use a random sample of 64 images to create our target embedding for the image CLIP guidance, the likelihood that animals were present on these 64 images is very high. This means that the target embedding, although mostly containing the “location concept”, also partly contains an “animal concept”. This means that the image CLIP guidance does not explicitly forbid the generation of new animals. Combined with optimizing the negative cross entropy loss, this leads to **adversarial animal insertions** at generation time, where a new animal of class  $\hat{y}$  appears alongside the original animal of class  $y$ , destroying the  $(\tilde{x}, y)$  alignment. In Fig. 11, we provide qualitative examples for this behaviour. To counter this behaviour, we choose  $\mathcal{L}_{\text{adv}}$  to be the “entropy” loss, or uncertainty loss. More precisely, this loss is equal to the cross entropy loss where the target label  $y$  is replaced by the soft label  $\tilde{y} = [\frac{1}{|\mathcal{Y}|}, \dots, \frac{1}{|\mathcal{Y}|}]$ , the uniform distribution over all classes. This loss explicitly encourages generations that either (1) do not contain new animals (2) contain new animals that are not accounted for in the label space  $\mathcal{Y}$ .

#### 7.5.3 Implementation details

**Adversarial optimization.** We describe here the parameters and settings used for optimization. If not precised, the same parameters were used for Adversarial Prompts and Guided Adversarial Prompts. As said in main paper Sec. 4.1, we optimize 4 prompts. Each prompt is composed of 10 placeholder tokens.

For optimization, we use a constant learning rate of 0.001, and a batch size of 8.

We use the “entropy” loss, described previously. For adversarial prompts, we train for 2000 steps. For guided adversarial prompts, we use CLIP guidance coefficient with  $\lambda_i = 10$  and  $\lambda_t = 0$  (i.e., no text guidance). We train for a total of 10000 gradient steps. However, we don’t optimize the adversarial loss for the first 2000 steps to allow the prompt to first converge to the target distribution region.

For adversarial prompts, to generate 4 different prompts, we simply change the seed. For guided (adversarial)

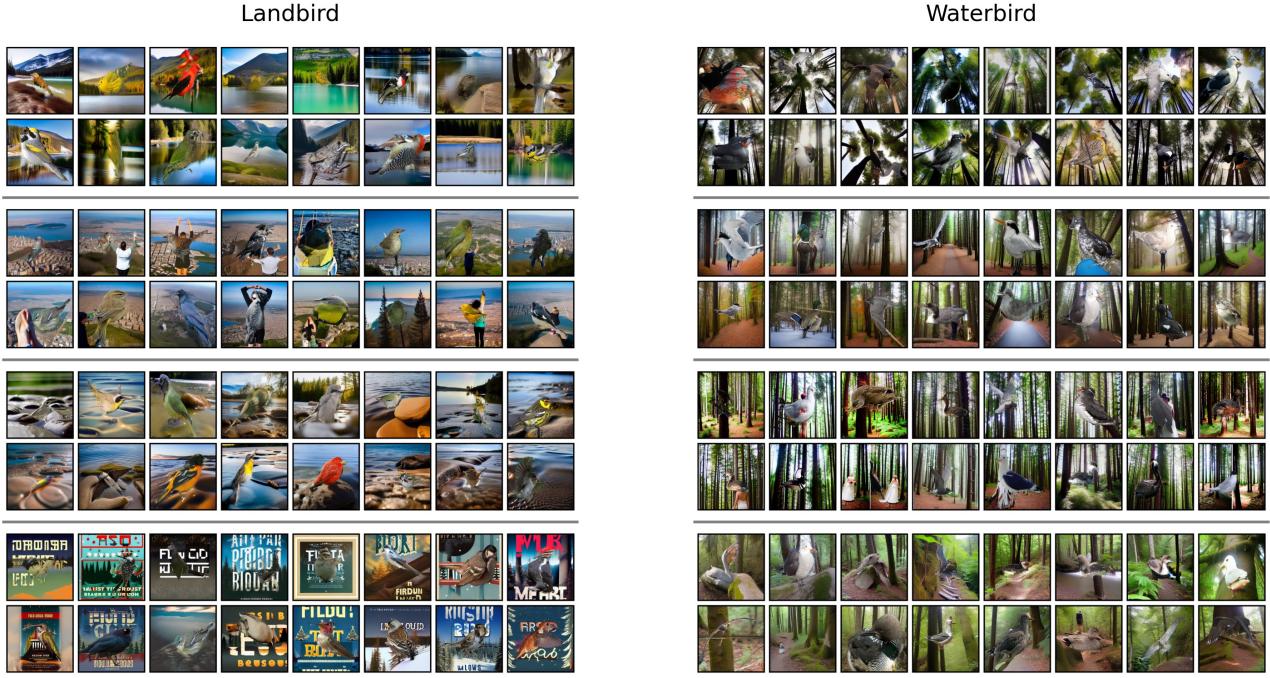


Figure 8. **Generation examples for Guided Adversarial Prompts.** Each column shows generations for the corresponding class. Each row in a column (separated by gray lines) shows generations for one found token. There are eight tokens in total, four for each class. **GAP** tend to generate landbirds on water (**left**) and of waterbirds on land (**right**), the combinations not present in the original training data (see Fig. 7)

prompts, each prompt is w.r.t a new location & time of the day of the test distribution.

**Training data.** The generation settings are the same as the ones used during adversarial optimization. For each target domain-guided adversarial prompt, (i.e. location & time of the day), the source images (used to condition the generation with an object mask and through SDEdit [50]) are only images that match the time of the day of the target domain used during generation. Furthermore, for each prompt, we only generate one image per source image.

For ALIA, for each prompt, we generate one image per source image, from the whole training dataset. For the generation settings, given we use a slightly different generation process (inpainting) compared to their original implementation, we search ALIA’s best-performing generation parameters (according to validation accuracy) over SDEdit strength [0.4, 0.5, 0.8] and guidance scale [5.0, 7.5]. We found the best-performing parameters for ALIA to be the same as the one reported by ALIA in their Github<sup>4</sup> i.e. SDEdit strength of 0.5 and guidance of 7.5.

<sup>4</sup><https://github.com/lisadunlap/ALIA>

**Finetuning.** The learning rate scheduler is a cosine scheduler, updated every epoch. The batch size is 128.

Our iWildCam pretrained model is taken from ALIA checkpoints<sup>5</sup>. ALIA trains the model from “scratch” (i.e. the model has never seen iWildCam data), for 100 epochs, on the combination of real + generated data. For our method, given we optimize the prompts based on a finetuned model feedback, it may not make as much sense to train the model from “scratch”. Thus, we also introduce the variant where the iWildCam pretrained model is finetuned on the combination of real + generated data for 20 epochs, where finetuning means that every layer, except the last, is frozen.

For a fair comparison, both training settings are tested for ALIA and our method. We found that ALIA worked best when training from scratch and our method worked best when using the finetuning setting.

Finally, in their iWildCam experiment, ALIA fixed the number of extra generated points to be used in combination with real data during training to 2224 images. For the sake of comparison, we adopt the same limit in our experiments,

<sup>5</sup><https://api.wandb.ai/files/clipinvariance/ALIA-iWildCamMini/brr7b3ks/checkpoint/ckpt-iWildCamMini-randaug-filtered-resnet50-0-0.001-0.0001/best.pth>

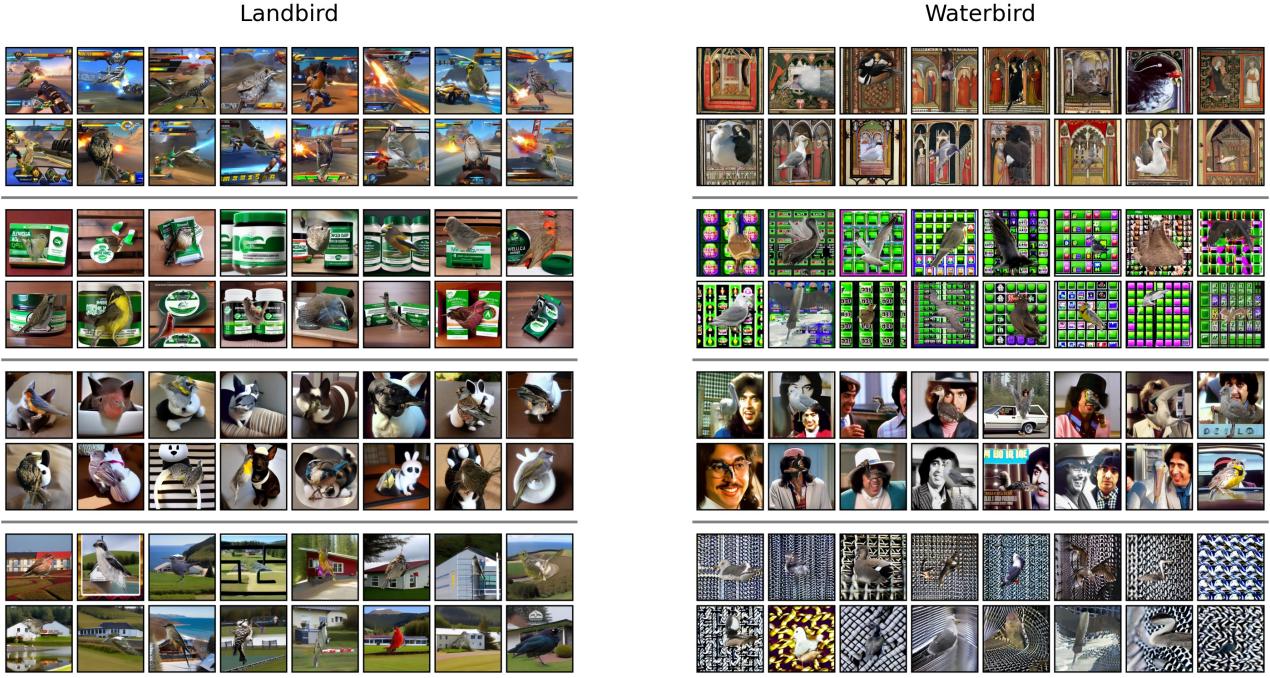


Figure 9. **Generation examples for Adversarial Prompts.** Each column shows generations for the corresponding class. Each row in a column (separated by gray lines) shows generations for one found token. There are eight tokens in total, four for each class. While AP finds tokens that fool the model, the generated images are different from the target distribution (land or water background).



Figure 10. **iWildCam Test Locations.** Random samples from the four target distributions. First row is LOCATION=1, day & night. Second row is LOCATION=2, day & night.



Figure 11. **Using the negative cross-entropy loss may lead to adversarial animal (e.g., elephants) insertions, destroying the alignment between  $\tilde{x}$  and  $y$ .** First row contains the original training images. The labels are [background, background, cattle]. Second row contains the corresponding generated samples using a guided adversarial prompt, optimized with negative cross-entropy loss as the adversarial loss.

with the added variant where the limit is 556 images, showcasing the data efficiency of our method.

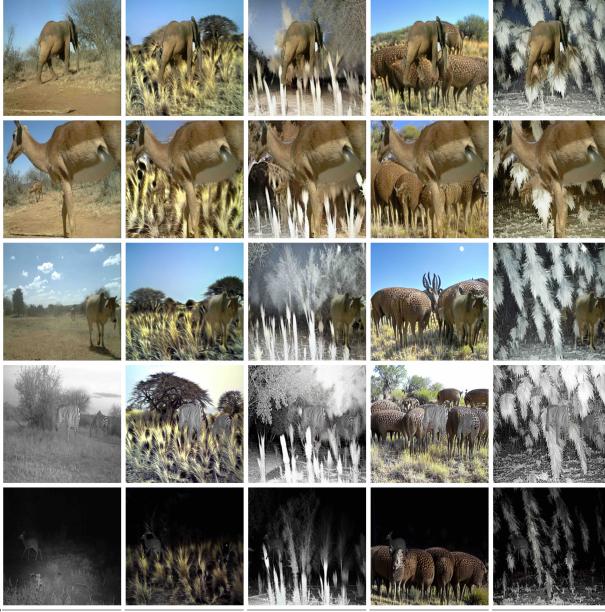


Figure 12. **Generations with the 4 guided adversarial prompts for iWildCam.** 1st column is the original image. Then from left to right, we have the guided adversarial prompts for LOCATION 1 during the day, during the night, and then LOCATION 2 during the day, during the night. SDEdit (strength 0.8) was used during the adversarial optimization and generation.

#### 7.5.4 Additional Qualitative Results.

In Fig. 12 and Fig. 13, we show a few generations using each of the 4 Guided Adversarial Prompts and Adversarial Prompts used in the iWildCam experiments.

#### 7.5.5 Using ViT model.

In Fig. 14, we repeat the iWildCam experiment from the main paper (Fig. 4) with a ViT-B-16 [16] model. Additionally, we provide qualitative results for generations from adversarial prompts optimized with a ViT-B-16 model in Fig. 15.

### 7.6. Image Guidance using Textual Inversion

In addition to the CLIP image guidance introduced in main paper Sec. 3.3, we also explore using Textual Inversion (TI) [22] as an image guidance mechanism. Similar to the CLIP guidance, we use a few images  $\{x_j\}$  from the target distribution. Now, instead of the similarity in a CLIP embedding space, we use the denoising loss between a generated image and one of the target images as in [22] (see Eq. (2)):

$$\mathcal{L}_{\text{TI}}(c_w) = \mathbb{E}_{x_j, z \sim \mathcal{E}(x_j), \epsilon \sim \mathcal{N}(0, I), t \sim U(0, 1)} \quad (4)$$

$$[\|\epsilon - \epsilon_\theta(z_t, t, c_w)\|_2^2], \quad (5)$$

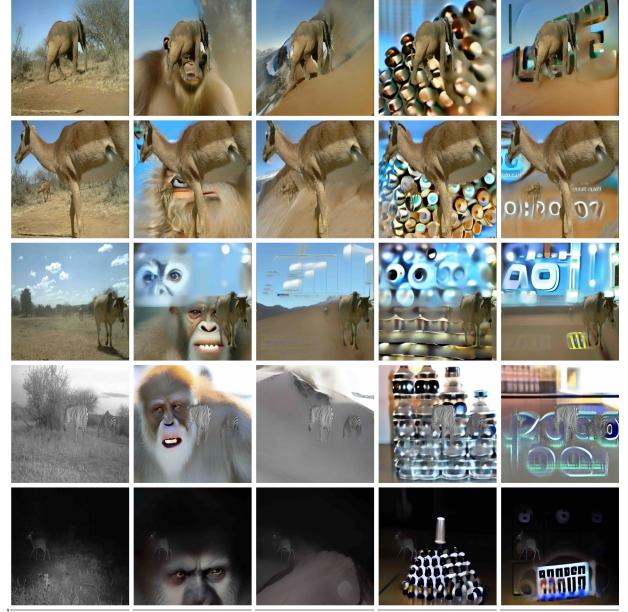


Figure 13. **Generations with the 4 adversarial prompts for iWildCam.** 1st column is the original image. Then, each column is an adversarial prompt initialized with a different seed. SDEdit (strength 0.8) was used during the adversarial optimization and generation.

where  $\mathcal{E}$  is the VAE [39] image encoder and  $\epsilon_\theta$  is the denoising UNet model from the Stable Diffusion model [61], and  $x_j$  is sampled randomly from the set of available target images.

We test the TI image guidance on the Waterbirds dataset. We use the guidance loss from Eq. (4) with the weight 1000 (we found lower values to result in generations less faithful to the target images) and randomly sample 50 (unlabeled) images from the validation split of the original Waterbirds dataset where both types of birds appear on both types of backgrounds. We keep other settings the same as for GAP and AP.

Fig. 16 shows that TI guidance works on par with or better than CLIP guidance on the Waterbirds dataset. We found, however, that the TI guidance does not result in faithful generations for iWildsCam dataset, and further investigations are needed.

### 7.7. Additional Analysis

We ablate hyperparameters like 1) using  $\ell_2$  or cosine loss for  $\mathcal{L}_{\text{CLIP}}$ , 2) different ways of incorporating guidance e.g., text or image with CLIP or textual inversion (T.I.).

The table shows the accuracy from different combinations of 1 and 2 on the Waterbirds dataset. Using  $\ell_2$  and text guidance worked best, thus, we used this setting for our results in Fig. 3. T.I. compares generations to the original images in the pixel space, and Image in the CLIP embed-

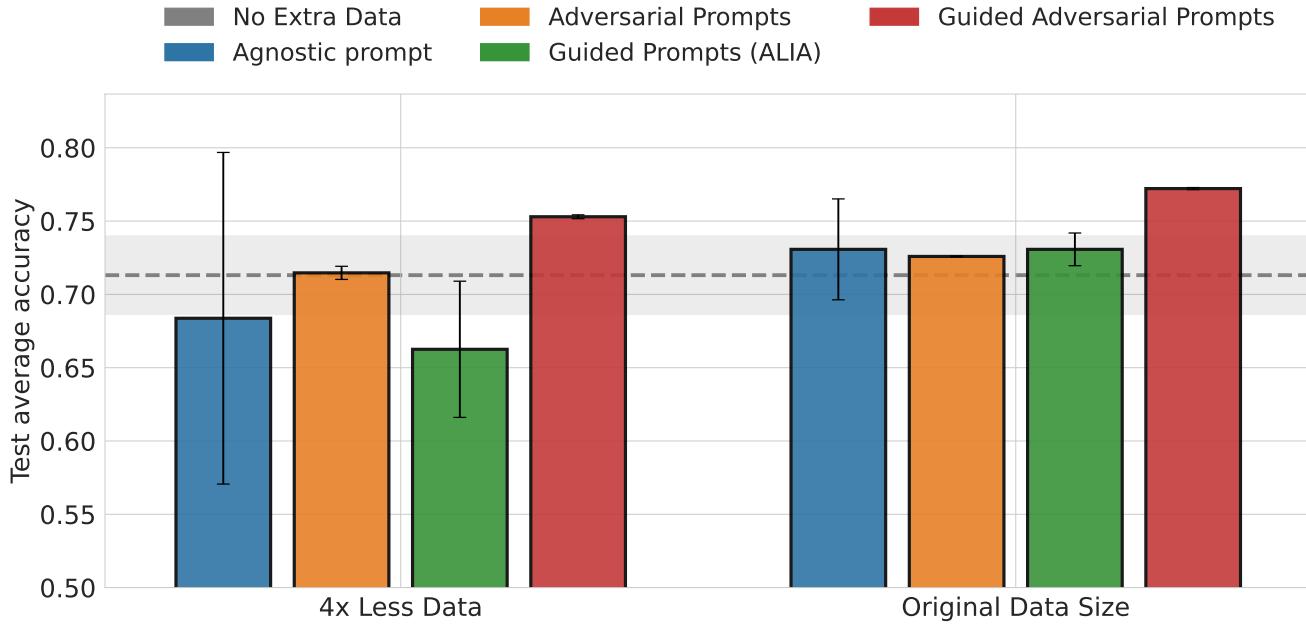


Figure 14. We train a ViT-B/16 model on the combination of the original training data and extra data generated using different types of prompts. We show the average accuracy on two iWildCam test camera trap locations. We run each experiment with three seeds and report the mean and standard deviation.

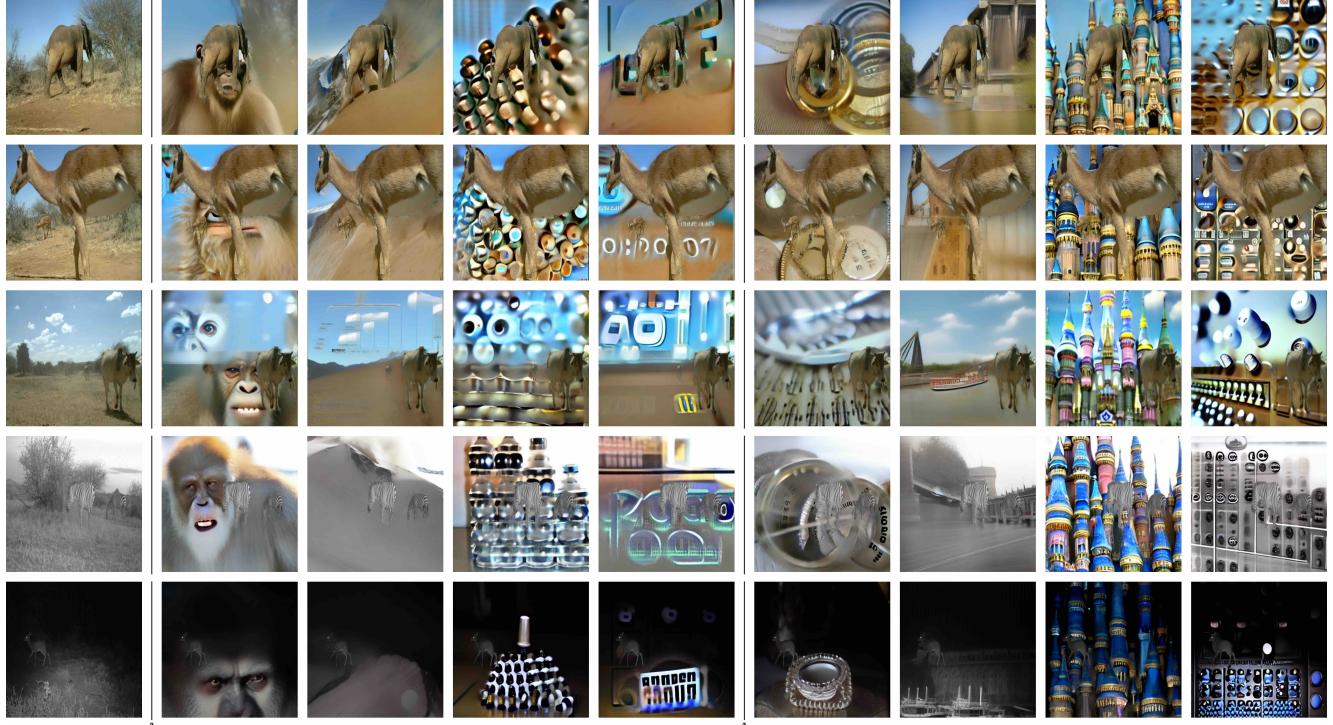
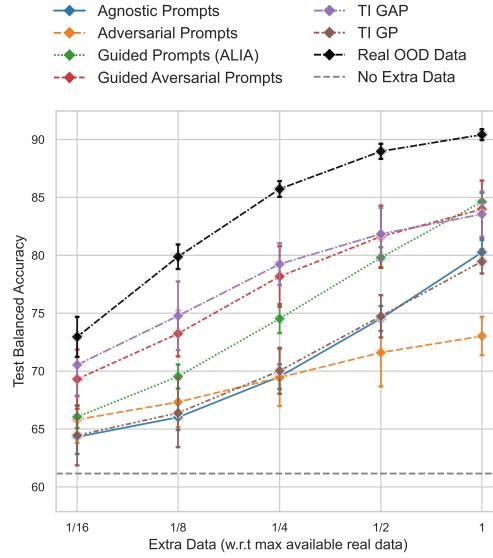


Figure 15. **A comparison of generations with adversarial optimization on different models.** First row is original image. Second to fifth row are generated using the four ResNet-based adversarial prompts. The four last rows are generated using the four ViT-based adversarial prompts.

ding space, resulting in different guidance mechanisms and, hence, performance.



**Figure 16. Results of using Textual Inversion image guidance.** We use the same plot as in Fig. 3. We add two additional methods using Textual Inversion image guidance described in Sec. 7.6. TI GAP is the Guided Adversarial Prompts with TI image guidance instead of the CLIP guidance. TI GP uses only TI guidance to find prompts. The TI guidance works on par with the CLIP guidance when used with adversarial optimization (TI GAP). However, using only TI guidance (TI GP) results in worse performance than using only text guidance with prompts found by the ALIA method [18].

Loss\Guide.	Text	Image	T.I.
$\ell_2$	84.0	77.3	83.6
Cosine sim.	82.5	72.7	—

**Table 5.** Results from different hyperparameters, namely, loss for CLIP guidance and how guidance is incorporated, on the Waterbird dataset.

## 8. Depth Estimation

### 8.1. Depth training details

**Adversarial Optimization.** The adversarial optimization was done with AdamW [46], learning rate of  $5.0 \times 10^{-4}$ , weight decay of  $1.0 \times 10^{-3}$ , and batch size of 8. The token embeddings at the start of optimization are randomly sampled from  $\mathcal{N}(\mu_{emb}, \sigma_{emb})$  where  $\mu_{emb}$  and  $\sigma_{emb}$  is the mean and standard deviation of all embeddings in the vocabulary. We set the early stopping threshold to 0.08 for the UNet model and 1.0 for the DPT model. **Note that these models were trained with different losses,  $\ell_1$  for the former and Midas loss [20] for the later.** Adversarial optimization is performed with the same loss as was used for training these models. One run takes approximately 30 mins on one A100. We perform a total of 32 runs, to get 32

Adversarial Prompts for the UNet model and 30 runs for the DPT model. As the DPT model was trained on Omnidata, which is a mix of 5 datasets, we have 6 runs for each dataset. Different number of placeholder tokens were also used for each run as suggested in Fig. 6b of the main paper. For the DPT model, we do 1, 8, 16 tokens runs for each dataset and also 3 runs with 32 tokens for each dataset. For the UNet model, 4 runs of 1, 8 and 16 tokens each and 16 runs of 32 tokens were used, to get a total of 32 prompts. We also use a reduced number of denoising steps during optimization i.e., 5, as we found it to be more stable.

**Guided Adversarial Optimization.** The CLIP guidance coefficient for text and image guidance is set to 1 and 5 respectively. For image guidance, we randomly sampled 100 images from the target distribution. For text guidance, we used target distribution’s name in the prompt, e.g., “fog” for the fog corruption from CC.

**Generation.** Generation is performed with the DDIM [72] scheduler and 15 sampling steps. We generate 80k images for the UNet model and 60k images for the DPT model for fine-tuning.

**Fine-tuning.** For fine-tuning, we optimize the UNet model with AMSGrad [60] with a learning rate of  $5.0 \times 10^{-4}$ , weight decay of  $2.0 \times 10^{-6}$  and batch size 128. For the DPT model, a learning rate of  $1.0 \times 10^{-5}$ , weight decay of  $2.0 \times 10^{-6}$  and batch size 32.

## 8.2. Additional Quantitative Results

**Performance of non-SD baselines.** In Tab. 6, we show the results for depth estimation for two additional baselines, deep augmentation [31] and style augmentation [23] that do not make use of generative models. Deep augmentation distorts a given image by passing it through an image-to-image model e.g., VAE [39], while perturbing its representations. Style augmentation involves applying style transfer to the original training images. They perform comparably to Adversarial Prompts.

## 8.3. Additional Qualitative Results

**Generations from all adversarial prompts & comparison of generations from different models.** We show the generations from all Adversarial Prompts from the UNet model, without SDEDit (Fig. 19), with SDEDit (Fig. 20), and multi-iteration (Fig. 21). Additionally, we provide the generations from two DPT models, allowing us to assess the difference the model feedback has on generations. The first DPT model was only trained on Omnidata (Fig. 22) and second was trained on Omnidata with augmentations from CC and 3DCC and with consistency constraints [79] (Fig. 23). The quantitative results in the paper were reported only on the former DPT model.

There does not seem to be obvious differences in the styles generated between the two DPT models. However,

Shift	U-Net				DPT	
	Taskonomy			Replica	Taskonomy	
	Clean	CC	3DCC	CDS	CC	3DCC
Control (No extra data)	2.35	4.93	4.79	5.38	3.76	3.42
Agnostic Prompts	2.47	5.03	4.17	5.30	4.06	3.58
Agnostic Prompts (Random)	2.38	4.96	4.11	5.14	3.88	3.51
Adversarial Prompts	2.49	4.36	4.02	5.12	3.40	3.28
Adversarial Prompts (SDEdit)	2.59	4.20	3.88	4.96	3.35	3.25
Deep Augmentation	2.42	4.24	3.70	5.01	2.83	3.70
Style Augmentation	2.42	4.15	3.85	5.16	2.80	3.10

Table 6. **Quantitative results on depth estimation.**  $\ell_1$  errors on the depth prediction task for a pre-trained U-Net and DPT model. (Lower is better. UNet losses are multiplied by 100 and DPT losses by 10, for readability. Note that the two models were trained with different losses, thus their numbers are not comparable to each other.). We evaluate on distribution shifts from Common Corruptions [29] (CC), 3D Common Corruptions [37] (3DCC) and cross-datasets (CDS), Replica [73]. The results from CC and 3DCC are averaged over all distortions and severity levels on Taskonomy. Our method is able to generate training data that can improve results over the baselines on several distribution shifts. Generations with AP (SDEdit) gives better results than AP under distribution shifts. Thus, also conditioning on the original image seems to be helpful for these shifts. For the DPT model, the trends are similar, AP performs better than the baselines. Deep augmentation and style augmentation do not make use of a generative model for generating extra data. They perform comparably to AP.

between the Adversarial Prompts from the UNet model with and without multi-iteration, the Adversarial Prompts from the latter seems to result in much more diverse styles.

## 8.4. Additional Analysis

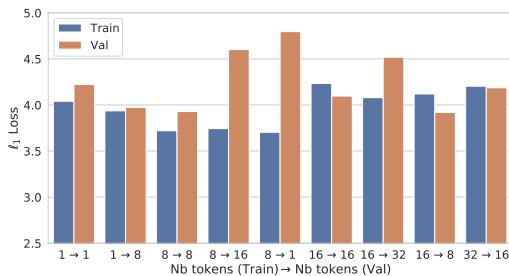


Figure 17. Generalization to generated images from similar or larger number of tokens. This plot shows the performance of the UNet model when trained on Adversarial Prompts with  $n$  number of tokens and tested on Adversarial Prompts with  $m$  number of tokens (denoted in the plot as  $n \rightarrow m$ ). With the exception of  $1 \rightarrow 1$  and  $1 \rightarrow 8$ , training on Adversarial Prompts with  $n$  tokens and testing on  $m, n \neq m$  results in higher loss than training and testing on the same number of tokens.

### 8.4.1 Running multiple iterations of adversarial optimization vs a single iteration

Here, we provide additional analysis for the multi-iteration experiments in Fig. 6b in the main paper. We optimize for 4 prompts in each iteration and noticed that if the number of placeholder tokens in a given prompt is kept fixed throughout the iterations, the optimization to find new Adversarial Prompts becomes more difficult. However, if we increase the number of tokens at each iteration e.g., 1 token per prompt for 1st, 8 per prompt for 2nd, etc, we are able to consistently find new Adversarial Prompts. Thus, we aim to investigate the generalization of a given model to different Adversarial Prompts, e.g., is a model more likely to generalize to Adversarial Prompts with the same number of tokens.

To perform this analysis, we generated data  $D_n, D_m$  using AP with  $n$  and  $m$  tokens per prompt respectively and measured the performance of a model fine-tuned on  $D_n$  on  $D_m$ .

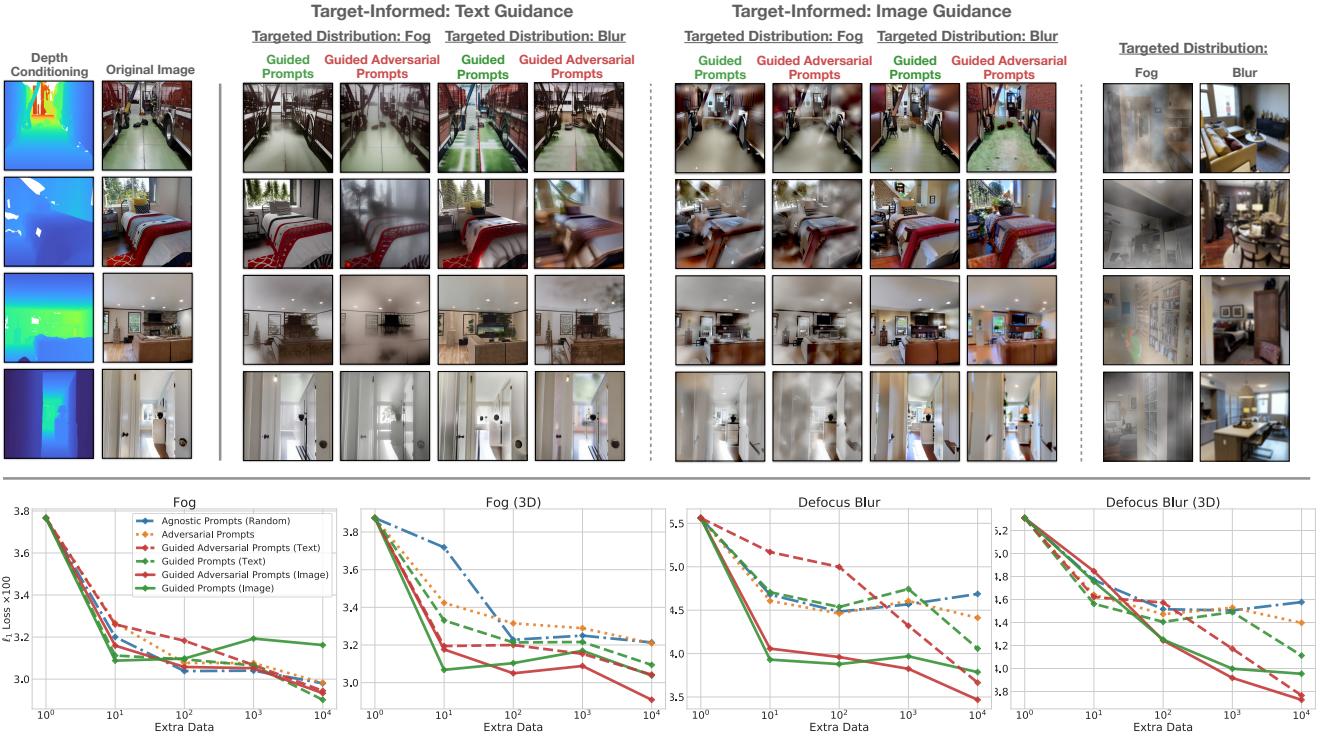
**Results for generalization to the same number of tokens.** In this setting,  $n = m$ , we use  $n = m \in \{1, 8, 16, 32\}$ . For every  $n$ , we construct  $D_n$  and  $D_m$  to be generated using 4 Adversarial Prompts. We fine-tune the model on  $D_n$  and validate both on  $D_n$  and  $D_m$  validation sets during the fine-tuning. The results are shown in Fig. 17.

**Results for generalization to different number of tokens.** In this setting,  $(n, m)$  are  $(1, 8), (8, 1), (8, 16), (16, 8), (16, 32), (32, 16)$  respectively. For every  $n$  we fine-tune on  $D_n$  and compute the validation loss on  $D_n$  and  $D_m$ . The results are shown in Fig. 17. As the loss for  $n = m$  tends to be more similar then when  $n \neq m$ , we chose to increase the number of tokens used per prompt in our multi-iteration setting.

### 8.4.2 CLIP image vs text guidance.

In Fig. 18, we compare the qualitative (top) and quantitative (bottom) differences in generations from text guidance and image guidance on defocus blur and fog. Note that image guidance uses sample (unlabelled) images from the corresponding target distribution that it is evaluated on, i.e., fog samples images from the fog corruption from the CC benchmark and fog (3D) samples images from the fog corruption of the 3DCC benchmark. If the target distribution name has (3D) appended to it, it is from the CC benchmark, otherwise it is from the 3DCC benchmark.

We observed some differences in the generations with text vs. image guidance (Fig. 18, top). Text Guided Prompts generates corruptions that are more realistic than image Guided Prompts. For example, fog gets denser



**Figure 18. A comparison of text and image guidance for two distribution shifts fog and blur.** The base model used here is the UNet model. **Top:** Generated images from [Guided Prompts](#) and [Guided Adversarial Prompts](#). All images were generated with SDEdit, strength 0.9, thus, they tend to look similar to the original image (2nd column). Using [Guided Prompts](#) alone for either text or image guidance results in generations with a *mild* fog or blur. With [Guided Adversarial Prompts](#), we get generations with *more severe* fog or blur. For image guidance, we sampled random (unlabelled) images with blur and fog from the Common Corruptions evaluation set. See the last two columns for some sample images. **Bottom:** The plots show the results from fine-tuning with text and image guidance, evaluated on fog and blur for CC and 3DCC. Note that the image guidance here uses (unlabelled) samples from the same target distribution that it was sampled on. In all cases, [Guided Adversarial Prompts](#) outperforms [Guided Prompts](#) with large enough extra data.

further away from the camera or around the floor when text Guided Prompts are used for generations. For image Guided Prompts, as it was guided by the image samples from CC where the corruption is applied uniformly over the image, it learns to also apply a more uniform corruption over the image.

Quantitatively, we observed that image guidance tends to perform the best across the target distributions, with large enough extra data (Fig. 18, bottom).

#### 8.4.3 Generalization of Adversarial Prompts to different models.

We show how adversarial generations from Adversarial Prompts found for one model are for another model in Tab. 7. The generations from Adversarial Prompts found for e.g., the UNet model result in the highest loss when evaluated on the UNet model. However, the generations from Adversarial Prompts from the DPT model also result in similar loss. Similar trends hold for the DPT model. Thus, Ad-

versarial Prompts found for one model are also able to result in high loss for another model.

AP from \ Eval on	Original data	UNet	DPT
UNet	2.55	7.63	5.39
DPT	1.76	7.17	6.46

**Table 7. Performance of a model on generated images from Adversarial Prompts from another model.** For the UNet model we report the  $\ell_1$  loss ( $\times 100$  for readability) and the DPT model, the Midas loss ( $\times 10$  for readability). The Adversarial Prompts attained from performing adversarial optimization on a UNet model and evaluated on the same model result in a loss of 7.63. Generations from the DPT model evaluated on the UNet model result in a loss of 7.17. Thus, the adversarial prompts found for one model seems to also be adversarial for another. We also report the loss on the original images for comparison.

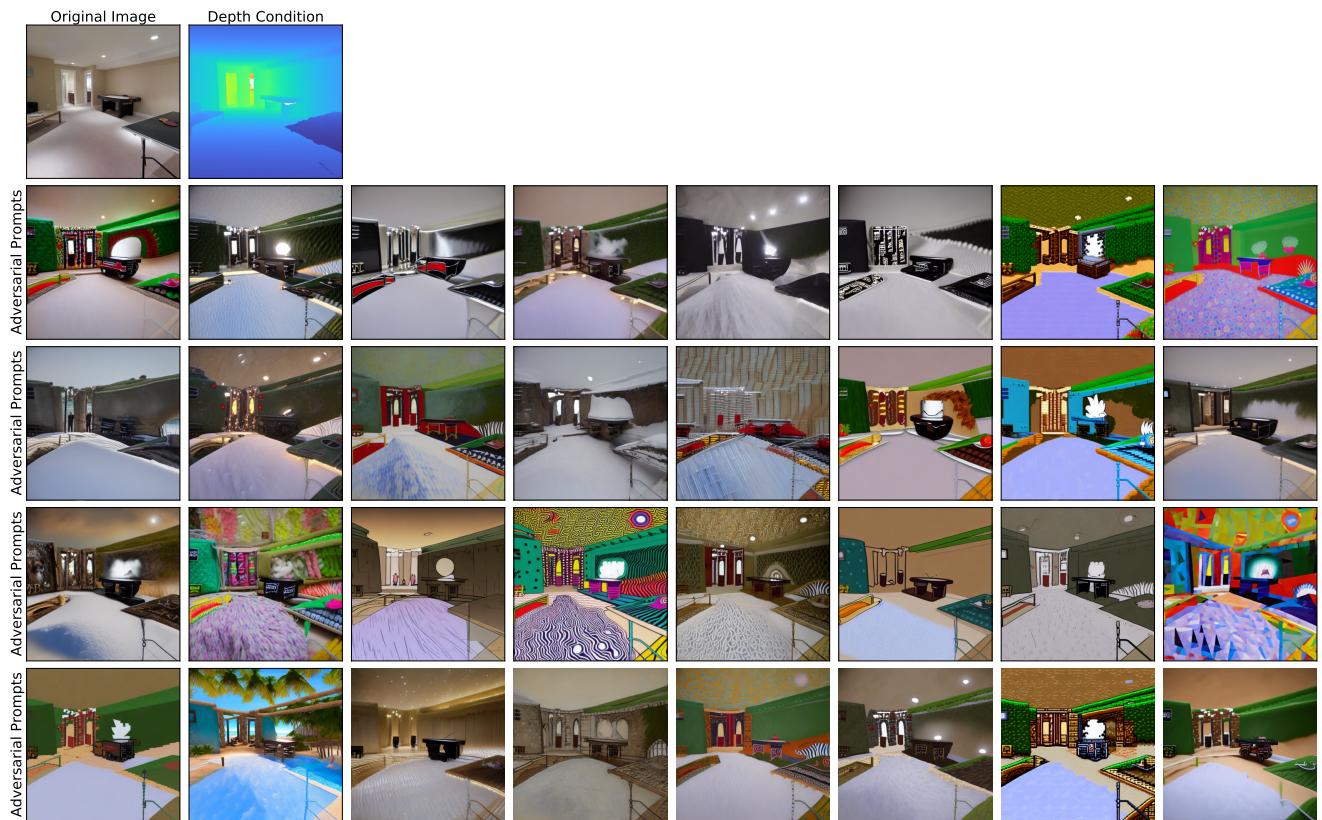


Figure 19. Generations from all Adversarial Prompts with the UNet model as the base model.

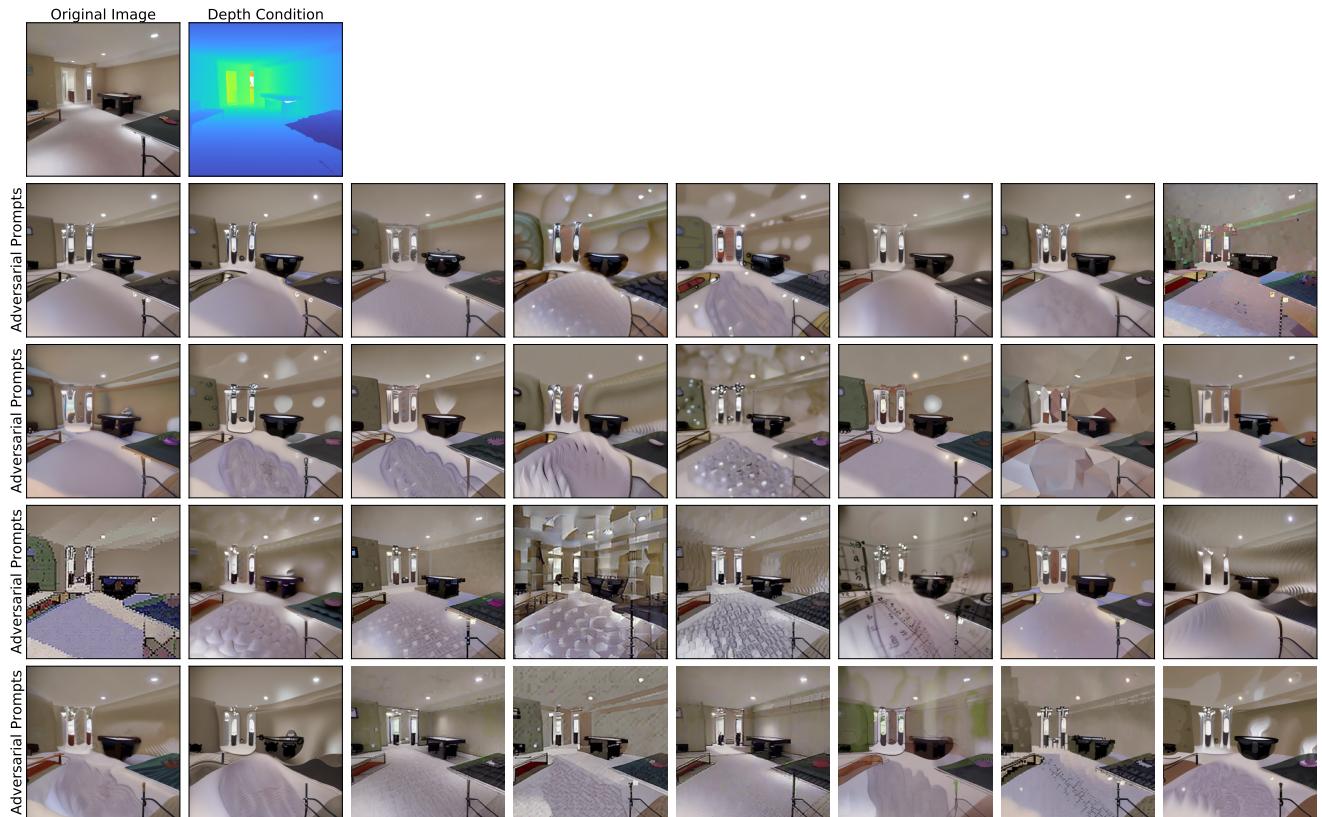


Figure 20. Generations from all Adversarial Prompts with the UNet model as the base model. SDEdit (strength 0.6) was used during the adversarial optimization and generation, thus, the generations look similar to the original image. Zoom in to see the different perturbations generated.

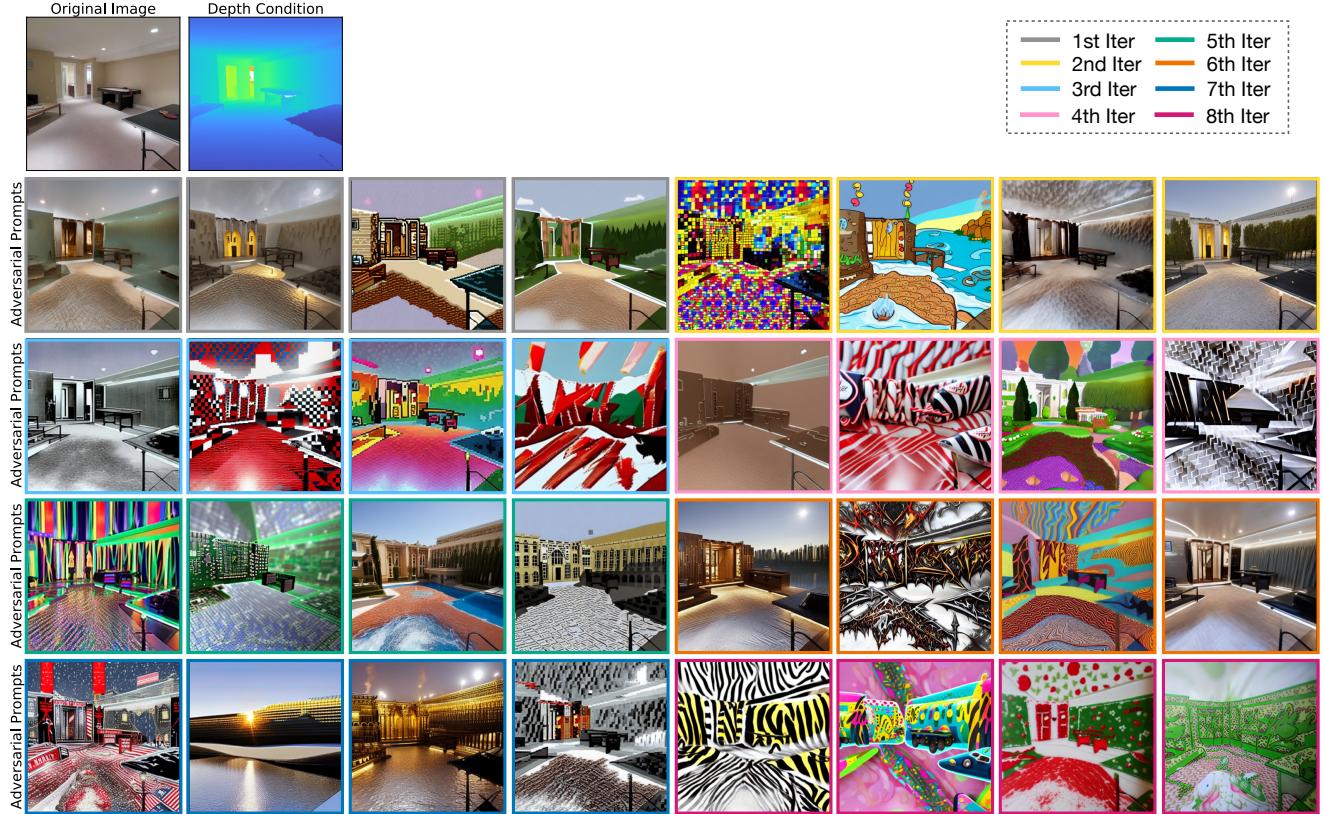


Figure 21. Generations from all Adversarial Prompts with the UNet model as the base model for the multi-iteration case i.e., multiple iterations of adversarial optimization, generation and fine-tuning. The colored borders denote the iteration number. Note that we set the early stopping threshold to be 0.1 for the first 3 iterations and 0.08 for the other iterations. We optimized for 4 prompts for each iteration, with an increasing number of tokens for each prompt.

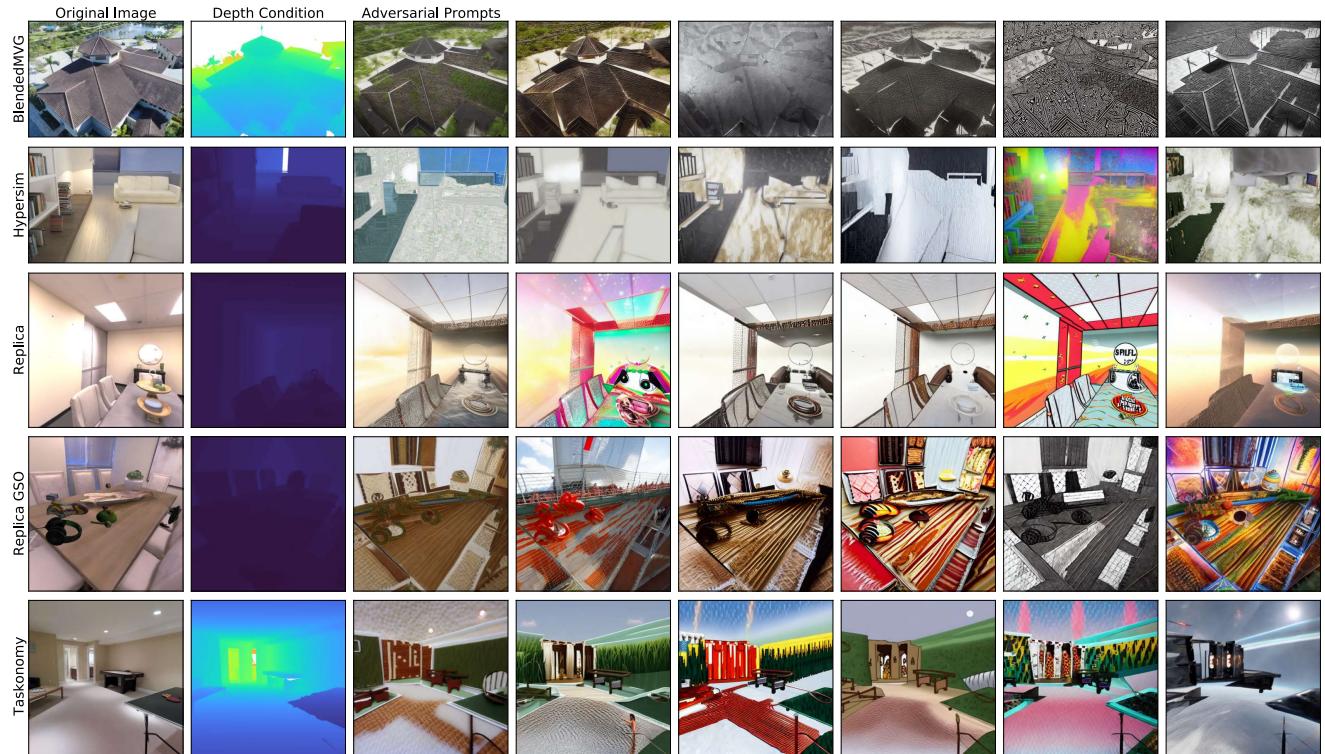


Figure 22. Generations from all Adversarial Prompts with the DPT model as the base model. The model was trained on Omnidata which consists of 5 datasets and we optimized for 6 Adversarial Prompts for each data. Each row shows the generation from the 6 different prompts for that dataset.

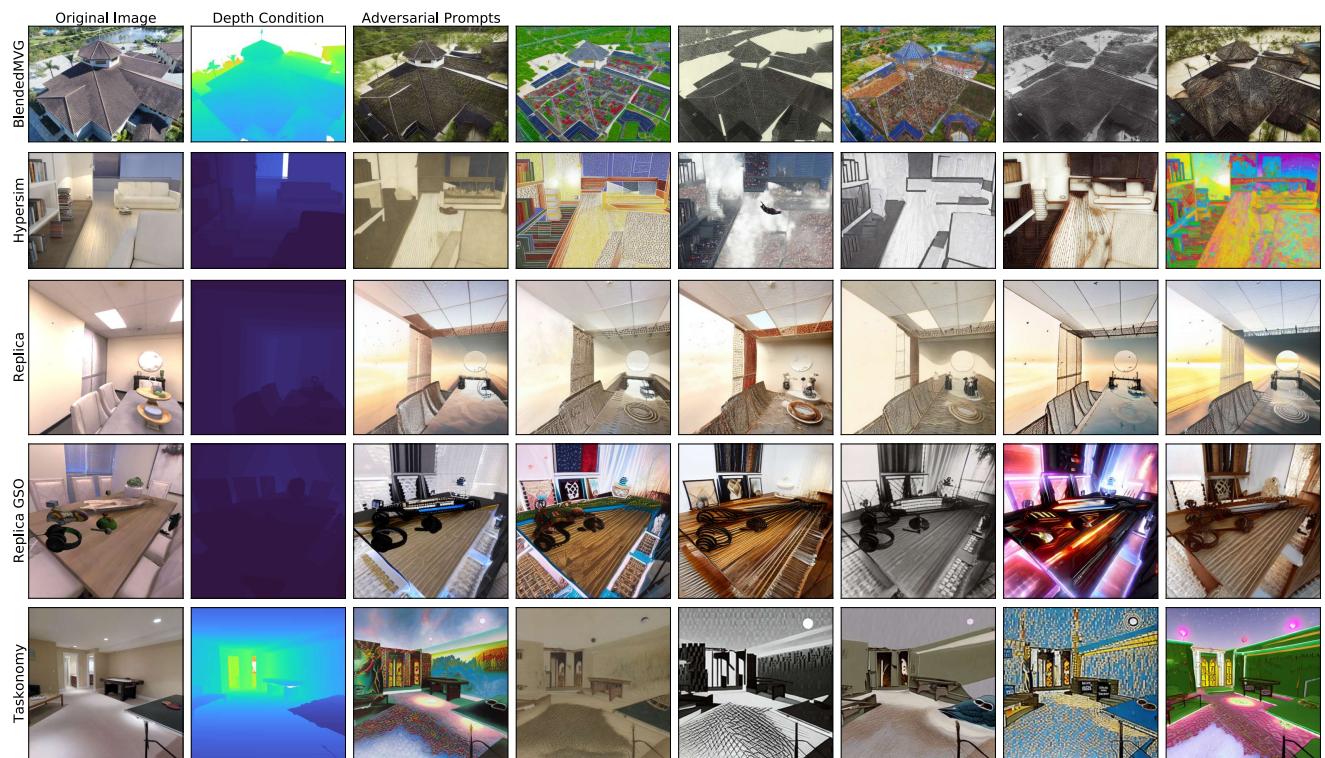


Figure 23. Generations from all Adversarial Prompts with a DPT model, also trained on Omnidata. However, this model was also trained with CC and 3DCC augmentations and consistency constraints.