

Fake it till you make it: Learning transferable representations from synthetic ImageNet clones

Mert Bulent Sariyildiz^{1,2}Karteek Alahari²Diane Larlus¹Yannis Kalantidis¹¹ NAVER LABS Europe² Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK

Abstract

Recent image generation models such as Stable Diffusion have exhibited an impressive ability to generate fairly realistic images starting from a simple text prompt. Could such models render real images obsolete for training image prediction models? In this paper, we answer part of this provocative question by investigating the need for real images when training models for ImageNet classification. Provided only with the class names that have been used to build the dataset, we explore the ability of Stable Diffusion to generate synthetic clones of ImageNet and measure how useful these are for training classification models from scratch. We show that with minimal and class-agnostic prompt engineering, ImageNet clones are able to close a large part of the gap between models produced by synthetic images and models trained with real images, for the several standard classification benchmarks that we consider in this study. More importantly, we show that models trained on synthetic images exhibit strong generalization properties and perform on par with models trained on real data for transfer. Project page: <https://europe.naverlabs.com/imagenet-sd/>

1. Introduction

The rise of (shallow) machine learning [16, 88] and later deep learning [28, 48, 83] has entirely changed the landscape of computer vision research over the past few decades, shifting some of the focus from *methods* to the *training data* itself. Datasets, initially of hundreds of images and dozens of classes [23, 24], have grown in size and complexity, and started becoming contributions in their own right. They have been fueling the progress of computer vision as much as, if not more than, the methods themselves. ImageNet [18], and mainly its ImageNet-1K [74] subset of about 1 million annotated images, has impacted the field in an unprecedented way. Yet, curating and annotating such a dataset comes at a very high money and labor cost.

The last couple of years have seen the rise of large and generic models, trained on data which is less curated but

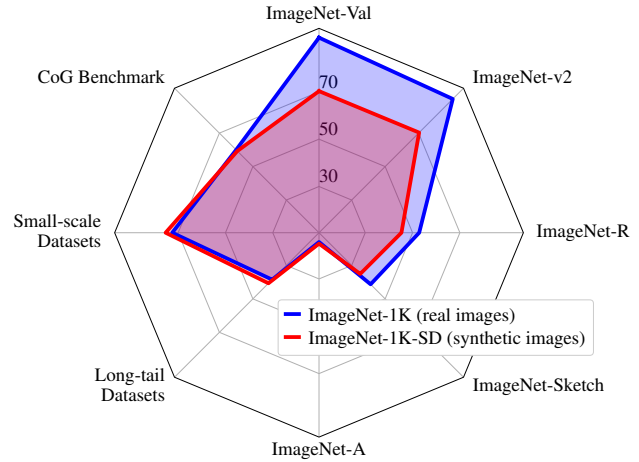


Figure 1. **ImageNet-1K vs ImageNet-1K-SD.** The blue polygon shows the performance of a model trained on ImageNet-1K. The red polygon depicts the performance of one trained on ImageNet-1K-SD, i.e., only on synthetic data generated with Stable Diffusion [73] using the class names of ImageNet-1K. We report top-5 accuracy for ImageNet test sets, and average top-1 for transfer tasks.

orders of magnitude larger. Those proved to be easily applicable, either directly, or combined with a tailored model, to a wide range of computer vision transfer tasks [39, 43, 68]. They have also been used beyond prediction tasks, e.g., for text-conditioned image generation. Models such as DALL-E [69] or Stable Diffusion [73] have demonstrated impressive image generation ability. They produce fairly realistic synthetic images and exhibit a high degree of compositionality.

Such generative models are trained on billion-scale datasets [79] composed of noisy image-text pairs scraped from the internet. Although training such models is out of reach for most institutions, a few of them have been made available to the community. Given the remarkable ability of these generative models, it is only natural to ask provocative questions such as: *Is there still a need for real images when training image prediction models?*

In this paper we explore this question through one of the most iconic computer vision datasets, ImageNet [18]. We study to which extent this dataset can be entirely replaced

by synthetic images when learning deep models. For this, we assume that we are provided with a set of classes, and the Stable Diffusion [73] model a generator that can produce realistic images from a textual prompt.

Our task is to learn an image classification model *from scratch* using a dataset composed only of synthetic images. We then evaluate the performance of this model on several datasets. First and foremost, we measure how well models and classifiers trained only on synthetic images recognize the training classes in real images from the standard ImageNet validation set. Then, we evaluate them on common datasets that test their resilience to domain shifts or adversarial examples, still for the ImageNet training classes. Finally, we consider several transfer learning scenarios where we measure the generalization performance of our models to novel classes. Fig. 1 summarizes the main results by comparing models trained on two equally sized set of images from the same set of classes, one real and one synthetic, on a number of these tasks. The gap is surprisingly narrow, especially for some of these scenarios.

To summarize, our contributions are threefold. First, we leverage Stable Diffusion [73] and generate synthetic ImageNet clones, *i.e.*, datasets with synthetic images for the ImageNet classes, using class names as prompts. We analyse the generated images, highlight important issues, and propose class-agnostic alterations to the basic prompt that reduce semantic issues and increase diversity. Second, we train classification models using different ImageNet clones and show that they can achieve 91.7% and 70.3% top-5 accuracy on ImageNet-100 and ImageNet-1K respectively. Finally, we evaluate the generalization capacity of our models. We show that their performance gap with models trained on real images is reduced when testing for resilience to domain shifts or adversarial examples. Moreover, we show that our models perform on par with models trained conventionally when testing on 15 transfer datasets.

2. Related work

2.1. Learning with synthetic data

Learning with synthetic data has become a standard way to create large amounts of labeled data for annotation heavy tasks, such as human understanding [67, 87], semantic segmentation [15, 76], optical flow estimation [20, 94] or dense visual alignment [66]. In most cases, this synthetic data requires access to 3D models and renderers [55], or to a simulator [72] with a physically plausible engine. Recent works propose pretraining on a database of synthetic fractal [44] or sinusoidal wave [84] images before fine-tuning the model using real images on a downstream task. In this study we use synthetic data to learn encoders and classifiers that can be used *out-of-the-box*, without the need for a subsequent fine-tuning step. Closest to our work, Kumar *et al.* [81] generate

synthetic OCT images to train a glaucoma detection model to be applied to real images. Here, we target synthetic clones of complex natural image datasets, *i.e.*, ImageNet-1K [74], and we use a *general-purpose* text-to-image generation model.

Synthetic ImageNet clones. Synthetic images for ImageNet classes have been used recently in a number of related works [3, 50, 70] based on class conditional Generative Adversarial Networks (GANs), such as BigGAN [7]. Besnier *et al.* [3] generate images for ten ImageNet classes and propose techniques to reduce the gap between models trained on generated images and real ones. Li *et al.* [50] synthesize five images for each ImageNet-1K class, together with their semantic segmentation annotations to automatically generate pixel-level labels at scale. Our work focuses on image-level classification, and uses a general-purpose text-conditioned generative model instead of ImageNet-1K class-conditioned GANs. It further offers a larger scale study with promising results on the full ImageNet-1K benchmark when training from 1.28 million synthetic images. Concurrent work [29] also synthesizes data for ImageNet-1K, but focuses on improvements on top of the CLIP [68] model or after fine-tuning.

Synthetic images as data++. Data sampled from generative models [26, 34, 69, 73] can be seen as data with added functionalities or “data++” [40]. Such data can be manipulated, interpolated or composed [12, 13, 41, 42] with dedicated operators in their latent space, and further used for counterfactual reasoning [51, 57, 61]. In this paper, we do not exploit these added functionalities. Our prompts consider a class at a time and do not leverage any interpolation nor the composition properties of synthetic data. Instead, we chose our complete pipeline, including the set of data augmentations, to be identical to the one we use for real images, to allow for a fair comparison.

Zero-shot learning and test-time view synthesis. Generative models have been used to extend models to new classes, or to create novel views at test time. Chai *et al.* [13] synthesize novel views for test-time ensembling by perturbing the latent code of a test image. Aiming at zero-shot recognition [96], Elhoseiny *et al.* [22] synthesize a classifier for any novel class given its semantic description (*e.g.*, textual or attribute-based), whereas others synthesize images [21, 27], or image *features* [49, 77] using such descriptions. Here we aim to learn encoders from scratch, and do not rely on models previously trained on real data.

2.2. Distillation of datasets and models

Knowledge distillation [8, 33] is a mechanism to transfer knowledge from a pretrained “teacher” model into a “student” one, and it usually requires images. Our approach can be seen as performing *image-free* distillation from a generic text-to-image generation model into a specific classification model. We assume no access to images to

distill from and, instead of distilling the visual encoder of the image generation model, inspired by recent works in NLP [53], we prompt a generation model to produce synthetic images and train a classifier with them.

Dataset distillation [11, 101], on the other hand, is a way of compressing a training set of real images into a smaller set of synthetic images such that after training a model on those, it performs as well as if it had been trained on the original set. However, one needs to tailor the generation process to a specific task, whereas in our case, we sample images from a task-agnostic generator.

Reconstructing images from model activations can be considered as another form of distillation. Earlier works reconstruct images from gradient-based features [90, 93] or CNN activations [54]. Since then many methods have tried to uncover the training data distribution as it is stored in the weights of a model [14, 99]. Instead of trying to recover the training distribution of the teacher image generation model, we use prompting to distill its knowledge for a specific image classification task.

3. Preliminaries

In this section, we first define the task we solve, *i.e.*, learning an image classification model when the training set of real images is replaced by an image generator, and training proceeds using only synthetically generated images. We then briefly describe Stable Diffusion [73], *i.e.*, the text-to-image generation model we use in this paper.

Task formulation. Our goal is to learn an image classification model given a set of class names \mathcal{C} and a text-to-image generator \mathcal{G} . This task is a variant of image classification where the fixed-size image training set is replaced by an image generator. The model we aim to learn consists of an encoder $\mathbf{z} = f_\theta(\mathbf{x})$ that maps an *image* \mathbf{x} into a vector representation $\mathbf{z} \in \mathbb{R}^d$, and a classifier $\mathbf{y} = q(\mathbf{z})$ that outputs a distribution \mathbf{y} over the N classes $c_i \in \mathcal{C}$, where $i = \{1, \dots, N\}$. We follow the common supervised learning setting [48, 74] and, unless otherwise stated, learn the encoder parameters θ together with the classifier q for the task. This model (encoder and classifier) is evaluated on the initial classification task, by applying it to real images (Sec. 5.1 and Sec. 5.2). We also evaluate the visual encoder in the context of several transfer learning tasks (Sec. 5.3).

Text-to-image with Stable Diffusion. We use the recent Stable Diffusion model [73] (SD) as text-to-image generator \mathcal{G} . SD is a denoising diffusion model [34] built around the idea of *latent diffusion*. The diffusion process is run on a compressed latent space for efficiency. An image encoder/decoder is used to interface the latent diffusion model with the pixel space. The generation process can be conditioned in many ways, *e.g.*, with text for text-to-image generation, or an image latent vector for image manipulation.

The text-to-image SD model consists of three main com-

ponents: i) an autoencoder whose visual encoder outputs a structured latent representation that is fed as input to the forward diffusion process and whose decoder is then used to convert the latent vectors back to pixels, ii) a denoising U-Net that runs the diffusion process, and iii) a text encoder, *i.e.*, similar to the one used by CLIP [68].

The text-to-image generation process takes a textual prompt p as input and generates an image $\mathbf{x} \in \mathbb{R}^{W \times H \times 3}$. Let $g(p)$ denote the generation function of model \mathcal{G} . Image \mathbf{x} is then given by $\mathbf{x} = g(p)$. In practice, the prompt p is first encoded via the text encoder and the text embedding is used as a conditioning vector for the latent diffusion process that runs for a number of steps. The latent representation is then provided to the decoder, which outputs the image \mathbf{x} .

There are two important parameters that control the quality and speed of text-conditioned diffusion; the number of diffusion steps and the coefficient that weights the textual conditioning vector. The former is linearly related to extraction time, while the latter provides an excellent way of controlling the visual diversity of generated images. The default values are 50 steps and guidance scale equal to 7.5.

Link to distillation. Since the generator is a model that internally encodes visual information, the image classification model we learn is essentially derived from \mathcal{G} . Under this formulation, and as discussed in Sec. 2, one can also see this task as text-guided, image-free knowledge distillation. Here we distill knowledge from a model of a very different nature, *i.e.*, a text-to-image generation model, to a purely visual encoder, for solving a specific task.

4. Generating synthetic ImageNet clones

For our study, we create clones of the ImageNet [18] dataset by synthesizing images depicting the classes it contains. We refer to all synthetic datasets of ImageNet classes that are created using Stable Diffusion as **ImageNet-SD**. Sec. 4.1 describes different ways of creating ImageNet-SD datasets starting from simply using the class name as the prompt. We then present generic, class-agnostic ways for tackling issues that arise with respect to semantics and diversity in Secs. 4.2 and 4.3, respectively. We present a few sample qualitative results in Fig. 2, with a more extensive set in the supplementary material.

4.1. Generating datasets using class names

In the absence of a training set of real images, we use the generator \mathcal{G} presented in the previous section to synthesize images for each class in the set \mathcal{C} . To do so, we need to provide the generator with at least one prompt per class. When used as an input, this class-conditioned prompt p_c triggers the generation of a synthetic image $\mathbf{x}_c = g(p_c)$ from class c . The simplest prompt one could think of is the class name *i.e.*, $p_c = "c"$. Although CLIP [68] uses $p_c = "a$

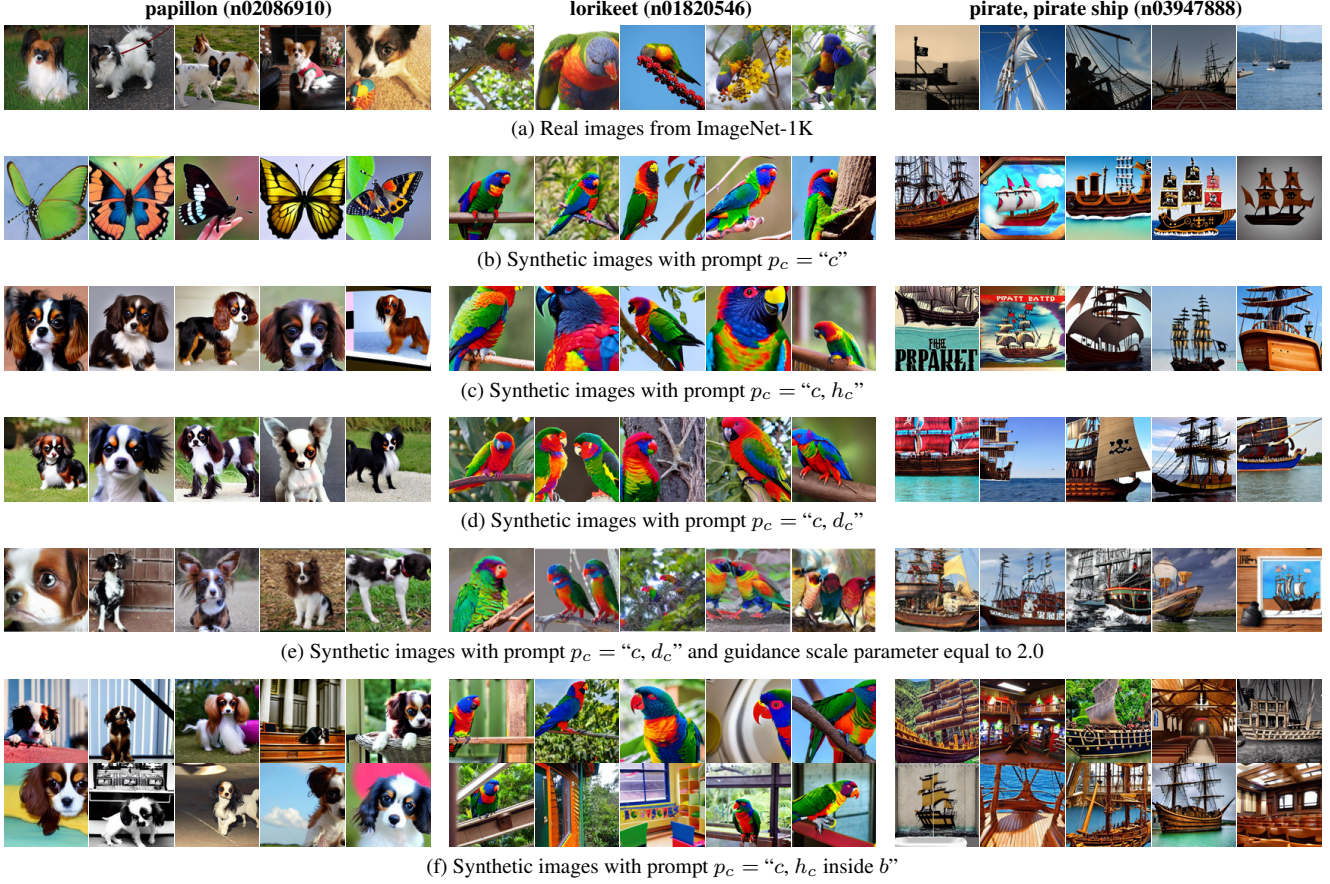


Figure 2. **Qualitative results.** (a) Real ImageNet images. (b)-(g) Synthetic ImageNet-SD images generated with different prompts. Despite high photo-realistic quality, some issues are noticeable for (b) such as i) semantic errors *e.g.*, for the class “papillon”, ii) lack of diversity, and iii) distribution shifts *e.g.*, towards cartoons for the “pirate” class. Such issues are addressed with more expressive prompts in (c)-(g).

photo of a c ” for their zero-shot experiments, using only the class name gives better results in our case.

Each class in ImageNet is associated with one or more *synsets*, *i.e.*, entities, in the WordNet [59] graph. We use the synset lemmas corresponding to each class as class-name prompt “ c ”, comma-separated if more than one. Fig. 2b shows random examples of images generated with such prompts. At first glance, one can appreciate the ability of the generator to create photo-realistic images given only a class name. In Sec. 5, we show that one can already obtain surprisingly good image classification results by simply training a model with this synthetic dataset.

Upon close inspection of the generated images, however, some issues become apparent: a) **semantic errors**: Images generated for some classes may capture the wrong semantics (*e.g.*, see the “papillon” class in Fig. 2b), b) **lack of diversity**: Generated images tend to look alike (an issue more apparent in the supplementary material, and c) **visual domain issues**: some classes tend to shift away from natural images towards sketches or art (*e.g.*, the “pirate ship” class in Fig. 2b). We discuss and address these issues in the following.

4.2. Addressing issues with semantics and domain

As mentioned earlier, by comparing the (real) images from ImageNet with the synthetic ones generated using only synset names as prompts, we observe that for some classes their semantics do not match. This is due to polysemy, *i.e.*, multiple semantic meanings or physical instantiations of the class names we used as prompt. We show one such case in the left-most column of Fig. 2b: the “papillon” images correspond to butterfly for our generated dataset, while the ImageNet synset contains images of the dog breed of the same name (see Fig. 2a).

To reduce this semantic ambiguity, we leverage once again the fact that class names correspond to WordNet [59] synsets. We augment the prompt for class name c with two additional elements provided by WordNet: a) The *hypernyms* h_c of the synset as defined by the WordNet graph, *i.e.*, the class name(s) of the parent node(s) of this class in the graph; and b) the *definition* d_c of the synset, *i.e.*, a sentence-length description of the semantics of each synset. In both cases, we append this information to the prompt, which becomes

$p_c = "c, h_c"$ and $p_c = "c, d_c"$ for hypernyms and definition, respectively.

Qualitatively, we observed that issues regarding the semantics of the most problematic classes are fixed, and so are, to some extent, issues related to visual domain mismatch. These are also visible in Figs. 2c and 2d: appending the hypernym ($h_c = "toy spaniel"$) or the description ($d_c = "small slender toy spaniel with erect ears and a black-spotted brown to white coat"$) of the class "papillon" in the prompt produces images with the dog breed as the main subject. Appending the hypernym ($h_c = "ship"$) or the description ($d_c = "a ship that is manned by pirates"$) of the class "pirate ship" results in more natural-looking images rather than illustrations, reducing the domain shift.

4.3. Increasing the diversity of generated images

Generating images using more expressive prompts, *e.g.*, by appending class hypernym or definition, not only reduces semantic errors, but also increases the visual diversity of the output images. This is visible, for example, in the "lorikeet" and "pirate ship" classes in Figs. 2c and 2d when compared to Fig. 2b: the pose and viewpoints are slightly more diverse. However, images still tend to display the class instance centered and in a prominent position. The real ImageNet images feature significantly more diversity, several different settings and backgrounds, and, in several cases, multiple instances of the same class (*e.g.*, see Fig. 2a).

Although class-specific prompt engineering is an appealing option, in this study we chose to remain generic, and to increase diversity in class-agnostic ways.

Reducing reliance on the textual prompt. The text-conditioned generation process of Stable Diffusion uses classifier-free diffusion guidance [35] which jointly trains both the conditional and unconditional diffusion models, and combines their estimates, resulting in a trade-off between sample quality and diversity. This trade-off is controlled by the guidance scale parameter, that has in practice been shown to produce high-quality images in the range of 6-9 (the default value is 7.5). Although visually detailed (see Figs. 2b to 2d), the resulting images lack diversity. We therefore experiment with reducing the guidance scale. Despite a small degradation in the visual quality of the generated images, setting the scale to 2.0 results in more diverse sets of images as shown in Fig. 2e.

Diversifying the background. We assume that class c can be seen "inside" a scene or background. To remain class-agnostic, we use all the scene classes from the Places dataset [102] as background for every class. We generate images for every possible combination of a class c and a scene $b \in \mathcal{B}$ from the set \mathcal{B} of 365 scenes in Places. We found that " c inside b " generally produces the best-looking results among a few prepositions we tried. However, we found that semantic and domain errors that arise from gen-

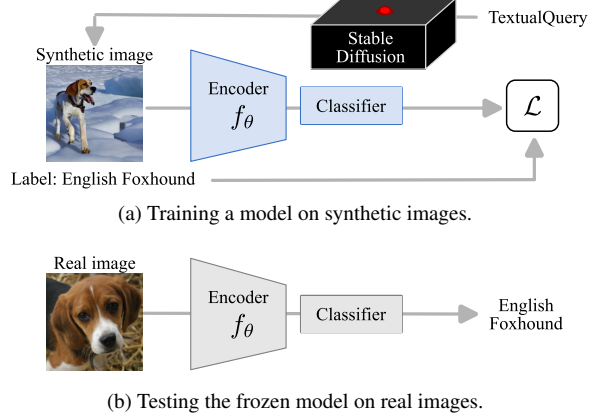


Figure 3. **Overview of our experimental protocol.** During training, the model has access to synthetic images generated by the Stable Diffusion model, provided with a set of prompts per class. During evaluation, real images are classified by the frozen model.

erating only using class name remained after specifying a background. We therefore build on top of the second simplest, but more semantically correct prompt variant, and use $p_c = "c, h_c$ inside $b"$ to generate images in diverse scenes and backgrounds. Although we do not consider this in our study, selecting backgrounds tailored for each class, *e.g.*, by matching class names to scenes using features from a text encoder, seems like a promising future direction.

Label noise and visual realism. Quite a few generated images, especially those with low guidance scale parameters or with random backgrounds (*e.g.*, see Figs. 2e and 2f) are not realistic, for example, the right-most image in the first column of Fig. 2e. When the prompt mentions a background, some images miss the foreground object completely (*e.g.*, see the bottom row in the middle column of Fig. 2f) or contain impossible combinations of objects and scenes. Yet, we see such noisy or unrealistic synthetic images as a way of adding stochasticity during the training process, similar to what strong non-realistic data augmentation achieves [25, 98]. In fact, it was recently shown [25] that diverse data augmentations, even when inconsistent with the data distribution, can be valuable (even more than additional training data) for out-of-distribution scenarios. Our experimental validation corroborates this claim.

5. Experiments

In this section we analyze the performance of image classification models learned using the different synthetic datasets constructed as described in Sec. 4. Due to the size of ImageNet-1K (roughly 1.3 million images), we perform most of our study on the smaller ImageNet-100 [85] dataset. This allows us to run multiple flavours of each synthetic dataset and to measure the impact of several design choices. Because ImageNet-100 is a randomly chosen subset of ImageNet-1K, spanning over 100 classes and

126,689 images, it preserves some important characteristics of ImageNet-1K such as its fine-grained nature.

We denote synthetic datasets for the two ImageNet subsets as **ImageNet-100-SD** (IN100-SD) and **ImageNet-1K-SD** (IN1K-SD), respectively.

Experimental protocol. We follow the protocol illustrated in Fig. 3. The generator \mathcal{G} is the Stable Diffusion [73] v1.4 model,¹ trained on the LAION2B-en dataset [79] and fine-tuned on a smaller subset filtered by an aesthetics classifier. During training, the generator is used to synthesize images for each class, which are then used for training the parameters of the encoder and the classifier. Unless otherwise stated, we create datasets of the exact same size as their real-image counterparts, *i.e.*, we generate the exact same number of images for every class as in the corresponding real dataset, maintaining any class imbalance.

We evaluate all the models on real images. When evaluating their performance over the ImageNet classes, we use both the encoder and the classifier learned during training to predict labels of real images for the 5 ImageNet datasets (Secs. 5.1 and 5.2). For transfer learning (Sec. 5.3), we use the pretrained encoder as a feature extractor, and learn a separate linear classifier on each of the 15 transfer datasets.

All our experiments use ResNet50 [28] as the encoder f_θ . Unless otherwise stated, we use 50 diffusion steps. We provide ablations for the diffusion steps and guidance scale as well as more implementation details in the supplementary material. We use multi-crop data augmentation [10], as it results in large performance gains for the models trained on ImageNet-SD (see supplementary for more details). Indeed, strong transformations have been shown to improve domain generalization [89], and to reduce the sim-to-real gap.

5.1. Results on ImageNet datasets

Evaluating different prompts on ImageNet-100. Tab. 1 compares the performance of models trained using variants of ImageNet-100-SD created with the different prompts presented in Sec. 4, for two different guidance scale values: 7.5 and 2. From the results for ImageNet-val and ImageNet-v2 (four left-most columns), we make the following observations: **(a)** Simply using the class name as a prompt and the default guidance scale (row 2), one can synthesize images and learn a visual encoder *from scratch* that already achieves *more than 70% Top-5 accuracy* (43% Top-1 accuracy) on ImageNet-100, a challenging 100-way classification task with many fine-grained classes. **(b)** Adding the hypernym or the definition from WordNet as part of the prompt (rows 3, 4) addresses some of the semantic and domain issues and translates into performance gains. **(c)** Generating objects on diverse backgrounds (row 5), even in a simple and class-agnostic way, gives the best results for the default guidance scale, reaching over 50% Top-1 and 76% Top-5 accuracy on

ImageNet-100. **(d)** Using a lower guidance scale value (2) leads to more diverse image sets (as discussed in Sec. 4.3) and translates into the best overall performance on ImageNet-100. **(e)** The exact formulation of the prompt has less impact when lowering the guidance scale; all the four prompt variants lead to similar performance as we see from rows 6-9.

Scaling the number of synthetic images. Unlike real datasets that are capped in the number of images they contain, ImageNet-SD has theoretically no size upper bound as one can generate images on demand. We therefore generated datasets which are $10\times$, $20\times$ and $50\times$ larger than ImageNet-100, using prompt $p_c = "c, d_c"$ (the best variant in Tab. 1, row 8) for the classes of ImageNet-100. From the last three rows of the top section in Tab. 1, we see that this brings gains of up to 8.5% in Top-1 accuracy on ImageNet-100, with our best model reaching 73.3% Top-1 (and 91.7% Top-5) accuracy. The gains are even more prominent for transfer learning, as we discuss in Sec. 5.3.

Results on ImageNet-1K. In the bottom part of Tab. 1 we report results on the very challenging 1000-way classification task of ImageNet-1K (IN-Val) that contains many fine-grained categories of mushrooms, birds and dogs [37]. We see that the model trained on our synthetic ImageNet-1K-SD dataset using the prompt composed of the class name and description ($p_c = "c, d_c"$) and using guidance scale 2 reaches 42.9% Top-1 and 70.3% Top-5 accuracy on the ImageNet-1K validation set. Although significantly lower than the results achieved by a model trained on the 1.3 million real images of ImageNet, we see that the synthetic dataset is able to at least partially capture the subtle clues needed to differentiate fine-grained classes. Similar observations can be made on ImageNet-v2 [71] (IN-v2).

5.2. Resilience to domain shifts

We investigate the performance of our models on three challenging evaluation sets for ImageNet-1K classes: ImageNet-Sketch [91] (IN-Sketch), ImageNet-R [31] (IN-R) and ImageNet-A [32] (IN-A). These datasets contain out-of-distribution images and their goal is to test resilience to domain shifts and adversarial images. Results are reported in the right-most columns of Tab. 1.

For ImageNet-100, we see from the top part of the table that a number of ImageNet-100-SD models *outperform* the model trained on real images for ImageNet-Sketch and ImageNet-R. The best ImageNet-100-SD model, *i.e.* the one trained with $50\times$ images, further rivals the baseline on ImageNet-A.

When it comes to a much harder classification task like the 1000 classes of ImageNet-1K, we see from the lower part of Tab. 1 that the same trend does not really hold. The ImageNet-1K-SD model trained on synthetic data lags behind in all cases when compared to the two models [64, 95] that are trained on the ImageNet-1K training set.

¹<https://huggingface.co/CompVis/stable-diffusion-v1-4>

| Training Dataset | R. Size | Scale | Prompt (p_c) / Model | IN-Val | | IN-v2 | | IN-Sketch | | IN-R* | | IN-A* | |
|------------------|---------|-------|--|--------|-------|-------|-------|-----------|-------|-------|-------|-------|-------|
| | | | | Top-1 | Top-5 | Top-1 | Top-5 | Top-1 | Top-5 | Top-1 | Top-5 | Top-1 | Top-5 |
| ImageNet-100 | 7.5 | – | ¹ <i>Baseline</i> | 87.4 | 96.8 | 82.5 | 95.1 | 39.1 | 58.9 | 58.4 | 79.1 | 25.6 | 68.7 |
| | | | ² $p_c = "c"$ | 43.1 | 70.7 | 45.4 | 70.7 | 29.9 | 53.5 | 51.7 | 75.3 | 8.8 | 38.4 |
| | | | ³ $p_c = "c, h_c"$ | 46.9 | 73.4 | 47.3 | 73.7 | 25.9 | 50.4 | 46.3 | 75.3 | 11.5 | 42.2 |
| | | | ⁴ $p_c = "c, d_c"$ | 47.9 | 74.2 | 49.1 | 74.9 | 24.7 | 49.2 | 41.2 | 71.5 | 12.2 | 38.5 |
| | | | ⁵ $p_c = "c, h_c \text{ inside } b"$ | 51.5 | 76.8 | 51.2 | 77.4 | 27.9 | 52.5 | 54.0 | 81.8 | 14.1 | 48.4 |
| | 2.0 | | ⁶ $p_c = "c"$ | 63.5 | 86.9 | 62.7 | 86.7 | 41.8 | 67.6 | 64.2 | 83.9 | 13.7 | 45.1 |
| | | | ⁷ $p_c = "c, h_c"$ | 63.4 | 87.1 | 63.5 | 86.5 | 39.2 | 66.7 | 61.9 | 85.1 | 14.9 | 49.1 |
| | | | ⁸ $p_c = "c, d_c"$ | 64.8 | 86.9 | 65.0 | 87.3 | 33.8 | 60.5 | 51.4 | 77.5 | 14.0 | 48.8 |
| | | | ⁹ $p_c = "c, h_c \text{ inside } b"$ | 63.1 | 85.7 | 62.0 | 85.0 | 38.7 | 65.5 | 64.0 | 87.2 | 21.9 | 63.1 |
| | 10× | 2.0 | ¹⁰ $p_c = "c, d_c"$ | 72.4 | 90.8 | 70.2 | 90.2 | 40.0 | 65.7 | 55.2 | 79.0 | 15.6 | 53.8 |
| | 20× | | ¹¹ $p_c = "c, d_c"$ | 72.4 | 91.4 | 71.4 | 90.7 | 38.4 | 63.9 | 56.9 | 81.5 | 17.8 | 55.0 |
| | 50× | | ¹² $p_c = "c, d_c"$ | 73.3 | 91.7 | 72.3 | 91.2 | 42.0 | 67.0 | 59.4 | 82.3 | 17.1 | 57.1 |
| ImageNet-1K | – | | ¹³ <i>PyTorch [58]</i> | 76.1 | 92.9 | 71.1 | 90.4 | 24.1 | 41.3 | 36.2 | 52.8 | 0.0 | 14.4 |
| | – | | ¹⁴ <i>RSB-A1 [95]</i> | 80.1 | 94.5 | 75.6 | 92.0 | 29.2 | 46.5 | 40.6 | 55.1 | 11.1 | 38.6 |
| ImageNet-1K-SD | 7.5 | | ¹⁵ $p_c = "c, d_c"$ | 26.2 | 51.7 | 26.0 | 51.4 | 9.5 | 22.1 | 15.9 | 32.0 | 2.2 | 10.1 |
| | 7.5 | | ¹⁶ $p_c = "c, h_c \text{ inside } b"$ | 30.1 | 55.6 | 29.8 | 55.3 | 11.9 | 27.1 | 23.5 | 43.1 | 3.4 | 13.2 |
| | 2.0 | | ¹⁷ $p_c = "c, d_c"$ | 42.9 | 70.3 | 43.0 | 70.3 | 16.6 | 35.1 | 26.3 | 45.3 | 3.6 | 15.1 |

Table 1. **Results on ImageNet datasets.** Top-1 and Top-5 accuracy on several ImageNet datasets, namely IN-Val (the ILSVRC-2012 validation set [74]), IN-v2 [71], IN-Sketch [91], IN-R [31] and IN-A [32]. In all cases, testing is done on real images. For the prompts, h_c (d_c) refers to the hypernym (definition) of class c provided by WordNet [59], while b to scene classes from Places 365 [102]. *IN-R and IN-A only cover a subset of the ImageNet-100 classes and we compute the reported metrics only on the common classes. Brick-colored scores denote performance higher than the models trained on real images. *Italics* denote results from models trained using real images.

| Training Dataset | Scale | Prompt (p_c) / Model | Aircraft | Cars196 | DTD | EuroSAT | Flowers | Pets | Food101 | SUN397 | iNat18 | iNat19 | Avg. |
|------------------|-------|---|----------|---------|------|---------|---------|------|---------|--------|--------|--------|------|
| – | – | ¹ Random Weights | 11.9 | 3.7 | 17.0 | 73.1 | 26.9 | 11.9 | 13.3 | 7.3 | 0.1 | 1.3 | 16.6 |
| ImageNet-100 | – | ² <i>Baseline</i> | 43.6 | 41.5 | 67.9 | 96.2 | 85.6 | 78.7 | 63.4 | 51.2 | 22.8 | 33.4 | 58.4 |
| ImageNet-100-SD | 2.0 | ³ $p_c = "c, d_c"$ (50× | 47.9 | 44.5 | 74.0 | 96.8 | 89.6 | 83.7 | 68.6 | 57.2 | 29.5 | 40.6 | 63.2 |
| ImageNet-1K | – | ⁴ <i>PyTorch [58]</i> | 48.9 | 49.9 | 72.1 | 96.2 | 89.3 | 92.3 | 71.2 | 60.5 | 35.5 | 41.5 | 65.7 |
| | – | ⁵ <i>RSB-A1 [95]</i> | 46.8 | 54.4 | 73.8 | 95.8 | 88.6 | 93.0 | 71.3 | 63.4 | 34.9 | 43.2 | 66.5 |
| ImageNet-1K-SD | 7.5 | ⁶ $p_c = "c, d_c"$ | 48.7 | 49.7 | 71.6 | 96.5 | 90.1 | 81.9 | 66.4 | 55.8 | 28.7 | 40.6 | 63.0 |
| | 7.5 | ⁷ $p_c = "c, h_c \text{ inside } b"$ | 49.6 | 47.4 | 72.1 | 95.9 | 89.3 | 87.2 | 67.7 | 59.5 | 30.8 | 41.4 | 64.1 |
| | 2.0 | ⁸ $p_c = "c, d_c"$ | 55.3 | 57.2 | 75.9 | 96.7 | 92.9 | 88.7 | 73.1 | 62.5 | 35.0 | 46.3 | 68.4 |

Table 2. **Top-1 accuracy on ten transfer learning datasets** for encoders trained on real and synthetic images. We treat encoders as feature extractors and train linear classifiers on top for each dataset. Brick-colored scores denote performance higher than the models trained on real images. We make the remarkable observation that representations from models trained on synthetic data can match the generalization performance of representations from models trained on millions of real images. *Italics* denote results from models trained using real images.

5.3. Transfer learning

In previous evaluations, we used pretrained models as a whole, *i.e.*, encoders together with classifiers, all trained on synthetic ImageNet datasets, and we directly applied those to predict the label of the (real) test images on the training classes. Here, we use a slightly different protocol. We evaluate the quality of the representations learned by our encoders alone, by using them as feature extractors and training linear logistic regression classifiers from scratch on top as done in transfer learning [46, 78].

We report results on 15 transfer datasets: (a) eight common small-scale datasets (Aircraft [56], Cars196 [47], DTD [17], EuroSAT [30], Flowers [60], Pets [63], Food101 [6], SUN397 [97]), (b) two long-tail datasets (iNat2018 [86] and iNat2019 [86]), and (c) the five datasets (“levels”) of the CoG benchmark [78]. We report Top-1 accuracy on the (real) test set of the small-scale and long-tail datasets in Tab. 2. In Fig. 1 and the supplementary, we present results on the CoG benchmark. We compare ImageNet-100-SD and ImageNet-1K-SD visual encoders obtained with some of our best prompts to baselines trained

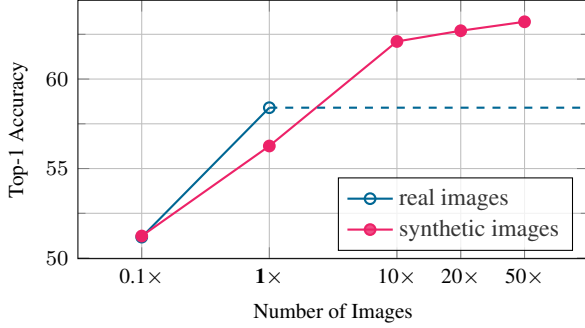


Figure 4. **Scaling the number of training images.** Average top-1 accuracy on 10 transfer datasets when training on ImageNet-100 using (1/10)-th to 50 \times images (relative to the real dataset size).

on ImageNet-100 and ImageNet-1K. What we observe is quite striking: On average, representations learned on purely synthetic images exhibit *generalization performance comparable to representations trained on thousands or millions of real images*. This suggests that synthetic images can be used to pretrain strong general-purpose visual encoders.

Following this transfer learning protocol, our best model achieves 70.4% Top-1 accuracy on ImageNet-1K (evaluation as part of the CoG benchmark, detailed in the supplementary material), significantly closing the gap to models trained on real data. This protocol differs from the one presented in Sec. 5.1 as it uses real images to train a linear classifier on top of the feature extractor trained only on synthetic images, hence results are not comparable with Tab. 1.

Scaling the number of synthetic images for transfer.

Fig. 4 reports transfer performance on the 10 datasets of Tab. 2, when varying the size of the training set. We see that generating 10 \times more images allows the ImageNet-100-SD model to outperform the model trained on real images, and the gains increase as we generate up to 50 \times .

6. Discussion

This section takes a step back and considers some of the implications from the analysis proposed in this paper.

Applicability beyond ImageNet. The process we followed to create ImageNet-SD requires minimal assumptions and can be applied to a wider set of classes. To disambiguate semantics, we only assume access to a short textual description of the class. This is generally easy to acquire even at a larger scale, *e.g.*, in semi-automatic ways from Wikipedia.

Scaling laws for synthetic data. Conceptually, there is no reason to restrict our approach to a finite dataset of synthetic images. We could devise a training process which sees each image only once [62].

Yet, despite this scaling potential, the quality of the resulting classifier is bounded by the expressivity of the generator and the concepts it can reliably reproduce. No matter how intriguing the promise of an “infinite dataset” via data generation might be, practical applications are bound by costs

linked to computation and storage, as well as the moderation of the content fueling this generator. The latter has strong implications we discuss next.

Data and model bias. Because of its pioneering role as a source of images to train generic models, and all it has done to advance the computer vision field, ImageNet and some of its bias has been under heavy scrutiny [19, 52]. Its synthetic counterparts have no reason to be immune to bias.

The main advantage of training with synthetic dataset is also its biggest flaw. Instead of manually curating and annotating a dataset, this process is outsourced to a text-to-image generator, whose training data is not always known. Our study is based on the text-to-image generator of Stable Diffusion (SD). SD is trained on LAION-2B [79], a dataset scraped from the internet and filtered in an automatic way using CLIP [68]. LAION has been shown to contain problematic content [5] and SD models to memorize at least part of the training set [9, 80]. Algorithmic bias is not only due to bias in the data [36], yet biased datasets lead to biased models and predictions [1, 75, 82]. Frameworks such as [38] could be considered to increase transparency and accountability.

On top of the bias in the data, the architecture itself constrains the generated images, and as such, propagates and potentially amplifies [4] existing bias. A major one that we have discussed earlier is the lack of diversity. An obvious corollary is the fact that stereotypes are reinforced. The options we have explored mitigate this issue to some limited extent, in that it improves classification results, but this issue is far from being solved. Finally, there are many societal implications of using such models to generate synthetic datasets for training computer vision models, and a more thorough and multi-disciplinary discussion is required.

7. Conclusions

In this paper, we study to which extent ImageNet, arguably the most popular computer vision dataset, can be replaced by a dataset synthesized by a text-to-image generator. Through an extensive study, we find that one can learn models that exhibit surprisingly good performance on fine-grained classification tasks like ImageNet-100 and ImageNet-1K without any class-specific prompting. However, the most important result of this study is the finding that models trained on synthetic data exhibit exceptional generalization capability that rivals with models learned with real images. We see this study as merely a first glimpse of what is now possible with the latest large models in terms of visual representation learning. We envision that similar approaches could be used to fine-tune or adapt models, using those synthetic datasets side-by-side with real ones.

Acknowledgements. This work was supported in part by MIAI@Grenoble Alpes (ANR-19-P3IA-0003), and the ANR grant AVENUE (ANR-18-CE23-0011).

References

- [1] Osman Aka, Ken Burke, Alex Bauerle, Christina Greer, and Margaret Mitchell. Measuring model biases in the absence of ground truth. In *AAAI/ACM Conference on AI, Ethics, and Society*, 2021. [8](#)
- [2] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proc. ICKDDM*, 2019. [12](#)
- [3] Victor Besnier, Himalaya Jain, Andrei Bursuc, Matthieu Cord, and Patrick Pérez. This dataset does not exist: training models from generated images. In *ICASSP*, 2020. [2](#)
- [4] Federico Bianchi, Pratyusha Kalluri, Esin Durmus, Faisal Ladhak, Myra Cheng, Debora Nozza, Tatsunori Hashimoto, Dan Jurafsky, James Zou, and Aylin Caliskan. Easily accessible text-to-image generation amplifies demographic stereotypes at large scale. *arXiv:2211.03759*, 2022. [8](#)
- [5] Abeba Birhane, Vinay Uday Prabhu, and Emmanuel Kahembwe. Multimodal datasets: misogyny, pornography, and malignant stereotypes. *arXiv:2110.01963*, 2021. [8](#)
- [6] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – Mining discriminative components with random forests. In *Proc. ECCV*, 2014. [7](#), [13](#), [15](#)
- [7] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *Proc. ICLR*, 2019. [2](#)
- [8] Cristian Buciluă, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proc. SIGKDD*, 2006. [2](#)
- [9] Nicholas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwal, Florian Tramèr, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. *arXiv:2301.13188*, 2023. [8](#)
- [10] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proc. ICCV*, 2021. [6](#), [12](#), [13](#)
- [11] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proc. CVPR*, 2022. [3](#)
- [12] Lucy Chai, Jonas Wulff, and Phillip Isola. Using latent space regression to analyze and leverage compositionality in GANs. In *Proc. ICLR*, 2021. [2](#)
- [13] Lucy Chai, Jun-Yan Zhu, Eli Shechtman, Phillip Isola, and Richard Zhang. Ensembling with deep generative views. In *Proc. CVPR*, 2021. [2](#)
- [14] Hanting Chen, Yunhe Wang, Chang Xu, Zhaohui Yang, Chuanjian Liu, Boxin Shi, Chunjing Xu, Chao Xu, and Qi Tian. Data-free learning of student networks. In *Proc. ICCV*, 2019. [3](#)
- [15] Yuhua Chen, Wen Li, Xiaoran Chen, and Luc Van Gool. Learning semantic segmentation from synthetic data: A geometrically guided input-output adaptation approach. In *Proc. CVPR*, 2019. [2](#)
- [16] Yunqiang Chen, Xiang Sean Zhou, and Thomas S Huang. One-class SVM for learning in image retrieval. In *Proc. ICIP*, 2001. [1](#)
- [17] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proc. CVPR*, 2014. [7](#), [13](#), [15](#)
- [18] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proc. CVPR*, 2009. [1](#), [3](#)
- [19] Emily Denton, Alex Hanna, Razvan Amironesei, Andrew Smart, and Hilary Nicole. On the genealogy of machine learning datasets: A critical history of ImageNet. *Big Data & Society*, 2021. [8](#)
- [20] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning optical flow with convolutional networks. In *Proc. ICCV*, 2015. [2](#)
- [21] Lisa Dunlap, Clara Mohri, Devin Guillory, Han Zhang, Trevor Darrell, Joseph E. Gonzalez, Aditi Raghunathan, and Anna Rohrbach. Using language to extend to unseen domains. In *Proc. ICLR*, 2023. [2](#)
- [22] Mohamed Elhoseiny, Babak Saleh, and Ahmed Elgammal. Write a classifier: Zero-shot learning using purely textual descriptions. In *Proc. ICCV*, 2013. [2](#)
- [23] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *IJCV*, 88, 2009. [1](#)
- [24] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. In *Proc. CVPR-W*, 2004. [1](#)
- [25] Jonas Geiping, Micah Goldblum, Gowthami Somepalli, Ravid Shwartz-Ziv, Tom Goldstein, and Andrew Gordon Wilson. How much data are augmentations worth? An investigation into scaling laws, invariance, and implicit regularization. In *Proc. ICLR*, 2023. [5](#)
- [26] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11), 2020. [2](#)
- [27] Sophia Gu, Christopher Clark, and Aniruddha Kembhavi. I can't believe there's no images! learning visual tasks using only language data. *arXiv:2211.09778*, 2022. [2](#)
- [28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016. [1](#), [6](#), [12](#)
- [29] 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? In *Proc. ICLR*, 2023. [2](#)
- [30] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. EuroSAT: A novel dataset and deep learning benchmark for land use and land cover classification. *JSTAEORS*, 2019. [7](#), [13](#), [15](#)
- [31] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proc. ICCV*, 2021. [6](#), [7](#), [13](#)

- [32] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proc. CVPR*, 2021. 6, 7, 13
- [33] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *Proc. NeurIPS-W*, 2014. 2
- [34] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proc. NeurIPS*, 2020. 2, 3
- [35] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. 5
- [36] Sara Hooker. Moving beyond “algorithmic bias is a data problem”. *Patterns*, 2021. 8
- [37] Minyoung Huh, Pulkit Agrawal, and Alexei Efros. What makes ImageNet good for transfer learning? *arXiv:1608.08614*, 2016. 6
- [38] Ben Hutchinson, Andrew Smart, Alex Hanna, Emily Denton, Christina Greer, Oddur Kjartansson, Parker Barnes, and Margaret Mitchell. Towards accountability for machine learning datasets: Practices from software engineering and infrastructure. In *ACM FAccT*, 2021. 8
- [39] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hananeh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. OpenCLIP, 2021. 1
- [40] Phillip Isola. When faking your data actually helps – learning vision from GANs, NeRFs, and noise, 2022. BMVC Keynote talk. 2
- [41] Ali Jahanian, Lucy Chai, and Phillip Isola. On the steerability of generative adversarial networks. In *Proc. ICLR*, 2020. 2
- [42] Ali Jahanian, Xavier Puig, Yonglong Tian, and Phillip Isola. Generative models as a data source for multiview representation learning. In *Proc. ICLR*, 2022. 2
- [43] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *Proc. ICML*, 2021. 1
- [44] Hirokatsu Kataoka, Kazushige Okayasu, Asato Matsumoto, Eisuke Yamagata, Ryosuke Yamada, Nakamasa Inoue, Akio Nakamura, and Yutaka Satoh. Pre-training without natural images. *IJCV*, 2022. 2, 14
- [45] Simon Kornblith, Ting Chen, Honglak Lee, and Mohammad Norouzi. Why do better loss functions lead to less transferable features? In *Proc. NeurIPS*, 2021. 14, 15
- [46] Simon Kornblith, Jonathon Shlens, and Quoc Le. Do better ImageNet models transfer better? In *Proc. CVPR*, 2019. 7, 12
- [47] Jonathan Krause, Jia Deng, Michael Stark, and Fei-Fei Li. Collecting a large-scale dataset of fine-grained cars. In *Proc. ICCV-W*, 2013. 7, 13, 15
- [48] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In *Proc. NeurIPS*, 2012. 1, 3
- [49] Michalis Lazarou, Tania Stathaki, and Yannis Avrithis. Tensor feature hallucination for few-shot learning. In *Proc. WACV*, 2022. 2
- [50] Daiqing Li, Huan Ling, Seung Wook Kim, Karsten Kreis, Sanja Fidler, and Antonio Torralba. BigDatasetGAN: Synthesizing ImageNet with pixel-wise annotations. In *Proc. CVPR*, 2022. 2
- [51] Shusen Liu, Bhavya Kailkhura, Donald Loveland, and Yong Han. Generative counterfactual introspection for explainable deep learning. In *GlobalSIP*, 2019. 2
- [52] Alexandra Sasha Luccioni and David Rolnick. Bugs in the data: How ImageNet misrepresents biodiversity. *arXiv:2208.11695*, 2022. 8, 18
- [53] Xinyin Ma, Xinchao Wang, Gongfan Fang, Yongliang Shen, and Weiming Lu. Prompting to distill: Boosting data-free knowledge distillation via reinforced prompt. In *Proc. IJCAI*, 2022. 3
- [54] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proc. CVPR*, 2015. 3
- [55] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *Proc. ICCV*, 2019. 2
- [56] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv:1306.5151*, 2013. 7, 13, 15
- [57] Chengzhi Mao, Augustine Cha, Amogh Gupta, Hao Wang, Junfeng Yang, and Carl Vondrick. Generative interventions for causal learning. In *Proc. CVPR*, 2021. 2
- [58] Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In *Proc. ACM-ICM*, 2010. 7, 13
- [59] George A Miller. Wordnet: A lexical database for English. *Commun. ACM*, 1995. 4, 7, 17
- [60] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Proc. ICCVGIP*, 2008. 7, 13, 15
- [61] Deniz Oktay, Carl Vondrick, and Antonio Torralba. Counterfactual image networks, 2018. 2
- [62] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113, 2019. 8
- [63] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *Proc. CVPR*, 2012. 7, 13, 15
- [64] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Proc. NeurIPS*, 2019. 6, 12, 13
- [65] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg,

- et al. Scikit-learn: Machine learning in Python. *JMLR*, 12, 2011. [12](#)
- [66] William Peebles, Jun-Yan Zhu, Richard Zhang, Antonio Torralba, Alexei A Efros, and Eli Shechtman. Gan-supervised dense visual alignment. In *Proc. CVPR*, 2022. [2](#)
- [67] Albert Pumarola, Jordi Sanchez-Riera, Gary P. T. Choi, Alberto Sanfeliu, and Francesc Moreno-Noguer. 3dpeople: Modeling the geometry of dressed humans. In *Proc. ICCV*, 2019. [2](#)
- [68] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proc. ICML*, 2021. [1](#), [2](#), [3](#), [8](#), [14](#)
- [69] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *Proc. ICML*, 2021. [1](#), [2](#)
- [70] Suman Ravuri and Oriol Vinyals. Classification accuracy score for conditional generative models. In *Proc. NeurIPS*, 2019. [2](#)
- [71] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In *Proc. ICML*, 2019. [6](#), [7](#), [13](#)
- [72] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *Proc. ECCV*, 2016. [2](#)
- [73] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proc. CVPR*, 2022. [1](#), [2](#), [3](#), [6](#), [14](#), [17](#)
- [74] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015. [1](#), [2](#), [3](#), [7](#), [13](#)
- [75] Hadi Salman, Saachi Jain, Andrew Ilyas, Logan Engstrom, Eric Wong, and Aleksander Madry. When does bias transfer in transfer learning? *arXiv:2207.02842*, 2022. [8](#)
- [76] Swami Sankaranarayanan, Yogesh Balaji, Arpit Jain, Ser Nam Lim, and Rama Chellappa. Learning from synthetic data: Addressing domain shift for semantic segmentation. In *Proc. CVPR*, 2018. [2](#)
- [77] Mert Bulent Sariyildiz and Ramazan Gokberk Cinbis. Gradient matching generative networks for zero-shot learning. In *Proc. CVPR*, 2019. [2](#)
- [78] Mert Bulent Sariyildiz, Yannis Kalantidis, Diane Larlus, and Karteek Alahari. Concept generalization in visual representation learning. In *Proc. ICCV*, 2021. [7](#), [12](#), [13](#), [16](#)
- [79] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. LAION-5B: An open large-scale dataset for training next generation image-text models. *arXiv:2210.08402*, 2022. [1](#), [6](#), [8](#)
- [80] Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Diffusion art or digital forgery? Investigating data replication in diffusion models. *arXiv:2212.03860*, 2022. [8](#)
- [81] Ashish Jith Sreejith Kumar, Rachel S. Chong, Jonathan G. Crowston, Jacqueline Chua, Inna Bujor, Rahat Husain, Eranga N. Vithana, Michaël J. A. Girard, Daniel S. W. Ting, Ching-Yu Cheng, Tin Aung, Alina Popa-Cherecheanu, Leopold Schmetterer, and Damon Wong. Evaluation of Generative Adversarial Networks for High-Resolution Synthetic Image Generation of Circumpapillary Optical Coherence Tomography Images for Glaucoma. *JAMA Ophthalmology*, 140(10), 2022. [2](#)
- [82] Ryan Steed and Aylin Caliskan. Image representations learned with unsupervised pre-training contain human-like biases. In *FAccT*, 2021. [8](#)
- [83] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proc. CVPR*, 2015. [1](#)
- [84] Sora Takashima, Ryo Hayamizu, Nakamasa Inoue, Hirokatsu Kataoka, and Rio Yokota. Visual atoms: Pre-training vision transformers with sinusoidal waves. *arXiv:2303.01112*, 2023. [2](#), [14](#)
- [85] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv:1906.05849*, 2019. [5](#)
- [86] Grant Van Horn, Oisín Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The iNaturalist species classification and detection dataset. In *Proc. CVPR*, 2018. [7](#), [12](#), [13](#), [16](#)
- [87] Gul Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *Proc. CVPR*, 2017. [2](#)
- [88] Andrea Vedaldi, Varun Gulshan, Manik Varma, and Andrew Zisserman. Multiple kernels for object detection. In *Proc. ICCV*, 2009. [1](#)
- [89] Riccardo Volpi, Diane Larlus, and Gregory Rogez. Continual adaptation of visual representations via domain randomization and meta-learning. In *Proc. CVPR*, 2021. [6](#), [13](#)
- [90] Carl Vondrick, Aditya Khosla, Tomasz Malisiewicz, and Antonio Torralba. HOGgles: Visualizing object detection features. In *Proc. ICCV*, 2013. [3](#)
- [91] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *Proc. NeurIPS*, 2019. [6](#), [7](#), [13](#)
- [92] Yizhou Wang, Shixiang Tang, Feng Zhu, Lei Bai, Rui Zhao, Donglian Qi, and Wanli Ouyang. Revisiting the transferability of supervised pretraining: an MLP perspective. In *Proc. CVPR*, 2022. [14](#), [15](#)
- [93] Philippe Weinzaepfel, Hervé Jégou, and Patrick Pérez. Reconstructing an image from its local descriptors. In *Proc. CVPR*, 2011. [3](#)
- [94] Yo whan Kim, Samarth Mishra, SouYoung Jin, Rameswar Panda, Hilde Kuehne, Leonid Karlinsky, Venkatesh Saligrama, Kate Saenko, Aude Oliva, and Rogerio Feris. How transferable are video representations based on synthetic data? In *NeurIPS Datasets and Benchmarks Track*, 2022. [2](#)

- [95] Ross Wightman, Hugo Touvron, and Hervé Jégou. Resnet strikes back: An improved training procedure in timm. In *Proc. NeurIPS-W*, 2021. **6, 7, 13**
- [96] Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *PAMI*, 41(9), 2018. **2**
- [97] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *Proc. CVPR*, 2010. **7, 15**
- [98] Zhenlin Xu, Deyi Liu, Junlin Yang, Colin Raffel, and Marc Niethammer. Robust and generalizable visual representation learning via random convolutions. In *Proc. ICLR*, 2021. **5**
- [99] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deep-inversion. In *Proc. CVPR*, 2020. **3**
- [100] Yaodong Yu, Kwan Ho Ryan Chan, Chong You, Chaobing Song, and Yi Ma. Learning diverse and discriminative representations via the principle of maximal coding rate reduction. In *Proc. NeurIPS*, 2020. **14, 15**
- [101] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *Proc. ICLR*, 2021. **3**
- [102] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *PAMI*, 2017. **5, 7, 18**

Contents

| | |
|--|-----------|
| A Implementation details | 12 |
| B Evaluation protocol | 12 |
| C Extended experimental results | 12 |
| C.1. Impact of data augmentation | 12 |
| C.2. Results on the ImageNet-CoG [78] benchmark | 13 |
| C.3. Analysis of the learned features | 14 |
| C.4. Impact of guidance scale and diffusion steps | 14 |
| C.5. Prefixing the prompt with domain identifiers | 14 |
| C.6. Additional scaling plots for synthetic data . . | 15 |
| C.7. Additional spider plots | 15 |
| D Extended qualitative results | 16 |
| D.1. Semantic errors | 16 |
| D.2. NSFW content | 17 |
| D.3. Misrepresentation of biodiversity | 18 |
| D.4. Semantic issues arising with backgrounds . . | 18 |
| D.5. Issues with diversity | 18 |
| D.6. Non-natural images | 18 |
| D.7. Varying the stable diffusion parameters . . . | 18 |

A. Implementation details

In all experiments the encoder f_θ is a ResNet50 [28] encoder, trained for 100 epochs (unless otherwise stated) with mixed precision in PyTorch [64] using 4 GPUs where batch norm layers are synchronized. We use an SGD optimizer with 0.9 momentum, a batch size of 256 and a learning rate linearly increased during the first 10% of the iterations and then decayed with a cosine schedule. Unless otherwise stated, we use the data augmentation pipeline from DINO [10] with 1 global and 8 local crops ($M_g = 1$ and $M_l = 8$). For Stable Diffusion we use 50 diffusion steps and a guidance scale factor of 7.5 for all experiments. We generate RGB images of size 512×384 .

B. Evaluation protocol

We evaluate our models in two ways. For the different ImageNet test sets, *i.e.*, datasets with images from the training classes (ImageNet-Val/v2/R/A/Sketch), we use the pretrained models as well as the classifiers we learn during pretraining with synthetic images. For the classification tasks on novel classes, *i.e.*, on the 10 small transfer datasets considered in Tab. 2 of the main paper plus the ImageNet-CoG benchmark in Appendix C.2, we freeze the pretrained encoder and train from scratch a new set of linear classifiers for each transfer task. The list of all datasets we use is given in Tab. 3.

For transfer learning evaluations, we follow the linear classification protocols from [46, 78]. More precisely, for each of the transfer datasets, we first extract image representations (features) from the pretrained encoders and then train linear logistic regression classifiers using these features. For the larger transfer datasets, *i.e.*, iNaturalist 2018 [86] and iNaturalist 2019 [86] datasets and the CoG levels, we train linear classifiers in PyTorch [64] using SGD, following [78]. For the remaining 8 smaller transfer datasets, we follow [46] and train classifiers using L-BFGS implemented in Scikit-learn [65]. In all cases, we resize the images with bicubic interpolation so that their shortest side is 224 pixels, and then take a central crop of 224×224 pixels. We tune hyper-parameters (learning rate and weight decay for the SGD optimizer, and regularization coefficient for the L-BFGS optimizer) using Optuna [2] over at least 25 trials. Code for evaluations can be found here².

C. Extended experimental results

C.1. Impact of data augmentation

We conducted some basic experiments to evaluate the impact of different data augmentation strategies when learning from synthetic datasets. In Tab. 4, we report the performance

²<https://github.com/naver/trex/tree/master/transfer>

| Dataset | # Classes | # Train samples | # Val samples | # Test samples | Val provided | Test provided |
|--|-----------|-----------------|---------------|------------------|--------------|---------------|
| <i>ImageNet test sets (training classes)</i> | | | | | | |
| ImageNet-Val [74] (IN-Val) | 1000 | — | — | 50000 | — | ✓ |
| ImageNet-v2 [71] (IN-v2) | 1000 | — | — | 3×10000 | — | ✓ |
| ImageNet-Sketch [91] (IN-Sketch) | 1000 | — | — | 50889 | — | ✓ |
| ImageNet-R [31] (IN-R) | 200 | — | — | 30000 | — | ✓ |
| ImageNet-A [32] (IN-A) | 200 | — | — | 7500 | — | ✓ |
| <i>Transfer tasks (novel classes)</i> | | | | | | |
| Aircraft [56] | 100 | 3334 | 3333 | 3333 | ✓ | ✓ |
| Cars196 [47] | 196 | 5700 | 2444 | 8041 | — | ✓ |
| DTD [17] | 47 | 1880 | 1880 | 1880 | ✓ | ✓ |
| EuroSAT [30] | 10 | 13500 | 5400 | 8100 | — | — |
| Flowers [60] | 102 | 1020 | 1020 | 6149 | ✓ | ✓ |
| Pets [63] | 37 | 2570 | 1110 | 3669 | — | ✓ |
| Food101 [6] | 101 | 68175 | 7575 | 25250 | — | ✓ |
| Pets [63] | 397 | 15880 | 3970 | 19850 | — | ✓ |
| iNaturalist 2018 [86] | 8142 | 437513 | — | 24426 | — | ✓ |
| iNaturalist 2019 [86] | 1010 | 265213 | — | 3030 | — | ✓ |
| CoG L_1 [78] | 1000 | 895359 | 223445 | 50000 | — | ✓ |
| CoG L_2 [78] | 1000 | 892974 | 222814 | 50000 | — | ✓ |
| CoG L_3 [78] | 1000 | 876495 | 218708 | 50000 | — | ✓ |
| CoG L_4 [78] | 1000 | 886013 | 221115 | 50000 | — | ✓ |
| CoG L_5 [78] | 1000 | 873630 | 218024 | 50000 | — | ✓ |

Table 3. **Datasets** we use for evaluating our models.

of models trained on the simplest variant of ImageNet-100-SD, *i.e.*, using the class name as the prompt, utilizing either PyTorch [58, 64] or DINO [10] augmentations. Although the gains for the real images are relatively small (less than one percent), the gains for ImageNet-100-SD are over 14%. We believe this shows two things: i) Synthetic images can benefit from the same augmentations as real images, and ii) these transformations are good for domain generalization. Indeed, strong transformations have been shown to improve domain generalization [89], and consequently can reduce the sim-to-real gap.

| Training Dataset | PyTorch [64] | DINO (+ Multi-crop) |
|-----------------------------|--------------|---------------------|
| ImageNet-100 (real) | 86.6 | 87.4 (↑ 0.80) |
| ImageNet-100-SD (synthetic) | 28.4 | 43.1 (↑ 14.6) |

Table 4. **Impact of data-augmentation** for models trained on real and synthetic datasets. Performance is measured on the validation set of ImageNet-100, *i.e.* on real images.

C.2. Results on the ImageNet-CoG [78] benchmark

We also evaluated our best ImageNet-SD model on the ImageNet-CoG benchmark introduced in [78] to measure concept generalization. This benchmark consists of evaluations on the set of training classes of ImageNet-1K (IN1K) and five “concept generalization levels”, *i.e.*, five IN1K-size

| Training Dataset | Prompt (p_c) / Model | IN1K | L_1 | L_2 | L_3 | L_4 | L_5 |
|------------------|--------------------------|------|-------|-------|-------|-------|-------|
| ImageNet-1K | PyTorch [58] | 75.8 | 67.8 | 63.1 | 58.9 | 58.2 | 52.0 |
| | RSB-A1 [95] | 79.8 | 69.9 | 65.0 | 60.9 | 59.3 | 52.8 |
| | DINO [10] | 74.8 | 71.1 | 67.2 | 63.2 | 62.6 | 57.6 |
| ImageNet-1K-SD | $p_c = “c, d_c”$ | 70.4 | 65.7 | 61.8 | 58.5 | 58.0 | 52.4 |

Table 5. **Top-1 accuracy on the ImageNet-CoG benchmark [78]**

We report performance for the best ImageNet-1K-SD model from Tab 2. of the main paper (with guidance scale equal to 2).

datasets of 1000 concepts each. These 5 concept generalization levels contain concepts from the full ImageNet-19K dataset which do not appear in IN1K. Moreover, they are ordered, *i.e.*, each containing concepts that are semantically further and further from the IN1K ones.

We follow the evaluation protocol presented in Appendix B and report Top-1 accuracy obtained on the test sets of these datasets in Tab. 5. We compare the performance of the best ImageNet-1K-SD model (from Tab. 2 of the main paper) to strong baselines trained on ImageNet-1K like the supervised RSB-A1 [95] and self-supervised DINO [10] models. We observe that on L_5 , which is the most challenging level, the performance of the representations learned on synthetic images is comparable to that of learned on real images. As we move towards L_1 , we see that the gap between these two models increases in favor of RSB-A1. Finally, after training classifiers (only) using the real images of IN1K, our model reaches 70.4% accuracy, significantly closing the

gap to even the most optimized models trained on real data like RSB-A1.

C.3. Analysis of the learned features

In this section, we analyze and contrast the *representations* obtained with models we trained using synthetic images to representations from models trained on real images. For this analysis, we used ImageNet-SD models for images that were generated using the default prompt guidance scale of Stable Diffusion, *i.e.*, 7.5. We perform our analysis for ImageNet-100 and using **four metrics**: i) Sparsity, ii) intra-class distance, iii) feature redundancy and iv) coding length. Note that we use the terms “representations” and “features” interchangeably.

We compare four different models trained on either real or synthetic data for the 100 classes of ImageNet-100: One model trained on real images, ImageNet-100-Real, two models trained on synthetic image sets of the same size obtained by using two different prompts: $p_c = “c”$ and $p_c = “c, h_c \text{ inside } b”$, and the ImageNet-100-SD-10x model, trained using ten times more images.

We perform these analyses on all the datasets listed in Tab. 3, except for the 5 ImageNet-CoG levels. For the sake of this study, we split them into three groups: i) ImageNet-100-Val/v2, ii) ImageNet-100-Sketch/A/R and iii) the 10 transfer datasets (long-tail and small-scale). For each pre-trained model and dataset, we extract features for either only the images in the test set (for the ImageNet test sets), or for all images (for the small transfer datasets). We then compute each of the four metrics separately on each dataset, and average them over all datasets in the same group. Before computing metrics, we ℓ_2 -normalize features.

Result analysis for each of the four metrics follows.

Sparsity. Inspired by [45], we compute feature *sparsity ratio*, *i.e.*, the percentage of feature dimensions close to zero with a threshold of 10^{-5} . We report sparsity ratios in Fig. 5a. We see that the sparsity ratio for the models trained on synthetic images increases as the “diversity” of a synthetic dataset increases, *i.e.*, we see gradual increase in sparsity scores from $p_c = “c”$ and $p_c = “c, h_c \text{ inside } b”$ to ImageNet-100-SD-10x. This observation aligns with their performance as well, *i.e.*, in the main paper we show that ImageNet-100-SD-10x performs best in general while $p_c = “c”$ performs worst. More interestingly, we see that ImageNet-100-Real, the model trained on real images, learns the most sparse representations.

Intra-class distance. In the main paper, we present simple ways to increase the diversity of synthetic images. Now we check if these efforts increase the variance of samples in the representation space. To do that, we compute the average ℓ_2 -distance between samples from the same class (*i.e.*, intra-class distance). We see in Fig. 5b that models trained with more diverse images indeed learn representations with

higher intra-class variance.

Feature redundancy. Following [92], we compute feature redundancy, *i.e.*, average pairwise Pearson correlation among dimensions. From Fig. 5c we see that the redundancy of features learned on real images increase more rapidly than the ones learned on synthetic images, as we move from ImageNet-100-Val/v2 towards out-of-domain or transfer datasets.

Coding length. To further investigate our observation on feature redundancy, we follow [100] and compute the average coding length per sample on each dataset (see Fig. 5d). We see that models trained on ImageNet-100-Real and ImageNet-100-SD-10x are comparable.

C.4. Impact of guidance scale and diffusion steps

In Fig. 6 we analyse the impact of the guidance scale and diffusion step hyper-parameters of Stable Diffusion [73]. As we discuss in the main paper, a lower guidance scale leads to more visual diversity and that is reflected of performance. Values of 1 to 3 all seem like a good choice. When it comes to the number of diffusion steps, values like 25 and (the default) 50 seem like a safe choice, with 25 being slightly worse, but requiring half the time to extract. Interestingly, using more steps seems to slightly hurt performance on the training classes. It is worth noting that transfer learning performance is surprisingly and consistently high for even 5 diffusion steps. This corroborates recent finding that training on complex but possibly semantically meaningless images like fractals [44] or sinusoidal waves [84] can provide a strong starting point for visual representations that generalize well.

C.5. Prefixing the prompt with domain identifiers

Handcrafted, dataset-level prompt engineering was used for the zero-shot experiments in the CLIP [68] paper. For example they use the prompt template “A photo of a c ” as default for classification tasks. For other fine-grained image classification datasets they go one step further and append “a type of {domain}” where {domain}={pet,food,aircraft} for datasets containing pet, food or aircraft classes.

In the main paper, instead presented automatic ways of clarifying the domain, *i.e.*, using extra information from WordNet for each class. In Tab. 6 we present some preliminary results when using generic prompt templates like “a photo of c ” and “an image of c ” as input to the Stable Diffusion v1.4 model. We found them to decrease performance for ImageNet-100.

| p_c | “ c ” | “a photo of c ” | “an image of c ” |
|------------|-------------|-------------------|--------------------|
| Top-1 Acc. | 64.8 | 59.5 | 58.3 |

Table 6. Top-1 Accuracy on ImageNet-100 when prepending the prompt with domain identifiers. Guidance scale is equal to 2.0.

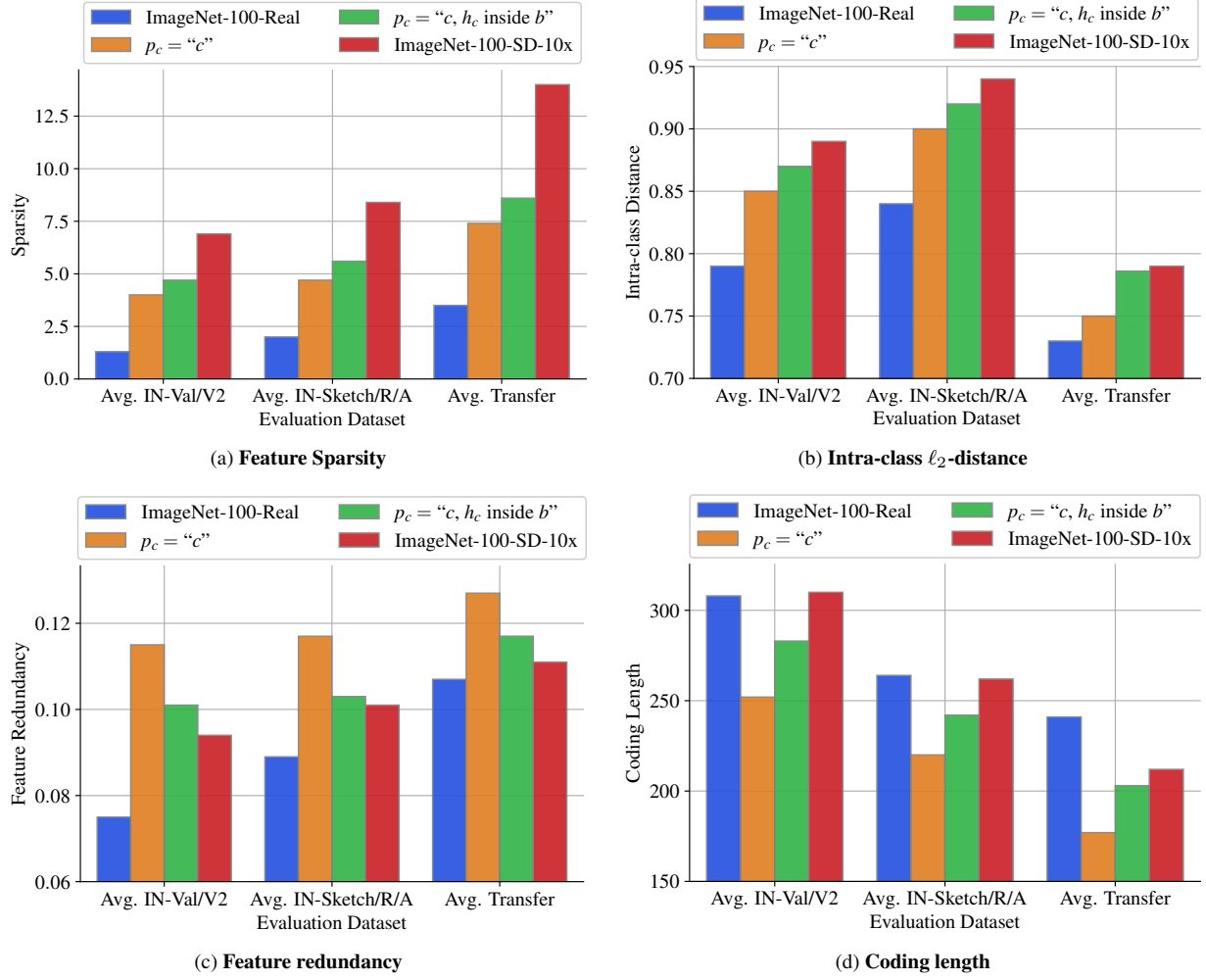


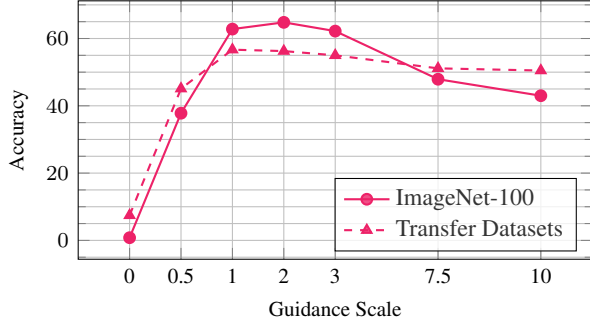
Figure 5. **Feature analyses** for models. We perform these analyses on top of features extracted from pretrained encoders f trained on either real or synthetic data for ImageNet-100 (training data is specified in the legends of the subfigures). For the purpose of this study, we use synthetic data generated with guidance scale equal to 7.5. *Sparsity* is measured by the percentage of dimensions close to zero [45]. *Intra-class ℓ_2 -distance* is the average pairwise ℓ_2 -distance between samples from the same class. These two metrics are computed on ℓ_2 -normalized features. *Feature redundancy* [92] is obtained by $\mathcal{R} = \frac{1}{d^2} \sum_i \sum_j |\rho(\mathbf{X}_{:,i}, \mathbf{X}_{:,j})|$, where $\mathbf{X} \in N \times d$ is a feature matrix containing N samples, each encoded into a d -dimensional representation (2048 in our case) and $\rho(\mathbf{X}_{:,i}, \mathbf{X}_{:,j})$ is the Pearson correlation between a pair of feature dimensions i and j . *Coding length* [100] is measured by $R(\mathbf{X}, \epsilon) = \frac{1}{2} \log \det(\mathbf{I}_d + \frac{d}{N\epsilon^2} \mathbf{X}^\top \mathbf{X})$, where \mathbf{I}_d is a d -by- d identity matrix, ϵ^2 is the precision parameter set to 0.5.

C.6. Additional scaling plots for synthetic data

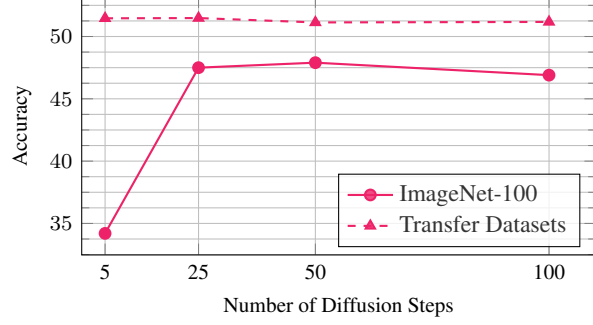
In Fig. 7 we report accuracy when training on ImageNet-100 using (1/10)-th to 50 \times images, relative to the real dataset size. Fig. 7a suggests that generating more images with basic prompts might not be enough, and that a performance leap will require advanced prompt engineering. We consider a study on scaling synthetic datasets is important, but beyond the scope of this paper. Note that Fig. 7b is also shown in the main paper and repeated here for completeness.

C.7. Additional spider plots

In Fig. 8 we show spider plots for the models trained on either real or synthetic data for ImageNet-100 and ImageNet-1K. In both cases, we show two plots which respectively report top-1 and top-5 accuracy for the ImageNet datasets, *i.e.*, ImageNet-Val/v2/R/A/Sketch. For transfer datasets and similar to the teaser figure in the main paper, we report top-1 accuracy averaged over the transfer datasets in each of the following three groups: **(a)** eight common small-scale datasets (Aircraft [56], Cars196 [47], DTD [17], EuroSAT [30], Flowers [60], Pets [63], Food101 [6], SUN397 [97]), **(b)** two

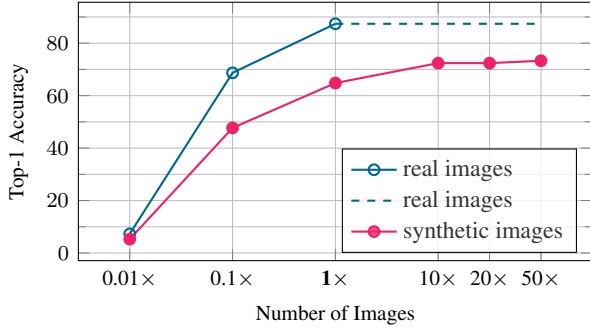


(a) Impact of the guidance scale parameter

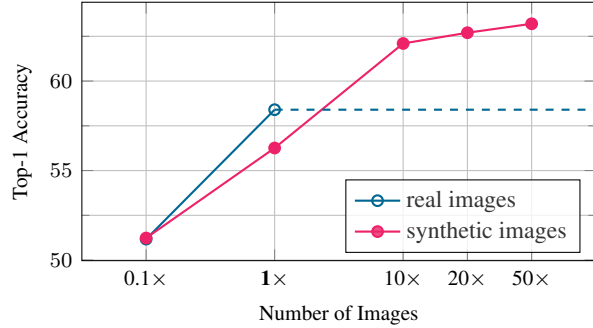


(b) Impact of the number of diffusion steps

Figure 6. **Impact of the guidance scale parameter and number of diffusion steps.** Top-1 Accuracy on ImageNet-100 and averaged over 10 transfer datasets for $p_c = "c, d_c"$. In the left plot, steps are set to 50, in the right plot guidance scale is 7.5.



(a) Top-1 accuracy on ImageNet-100.



(b) Top-1 accuracy on the 10 transfer datasets from Tab. 2 of the main paper.

Figure 7. **Scaling the number of training images.** Accuracy when training on ImageNet-100 using $(1/10)$ -th to $50\times$ images (relative to the real dataset size). Fig. 7b is also shown in the main paper.

long-tail datasets (iNat2018 [86] and iNat2019 [86]), and (c) the five datasets (“levels”) of the CoG benchmark [78].

D. Extended qualitative results

In this section, we provide additional qualitative results. First we show random images for *all* ImageNet-100 classes from three datasets: ImageNet-100-Val (real images) and two ImageNet-100-SD datasets generated by the prompts $p_c = "c"$ and $p_c = "c, h_c \text{ inside } b"$. Then we discuss in more detail several types of issues that we observed in these synthetic images. Unless otherwise stated, the guidance scale used is 7.5.

Qualitative results for *all* ImageNet-100 classes.

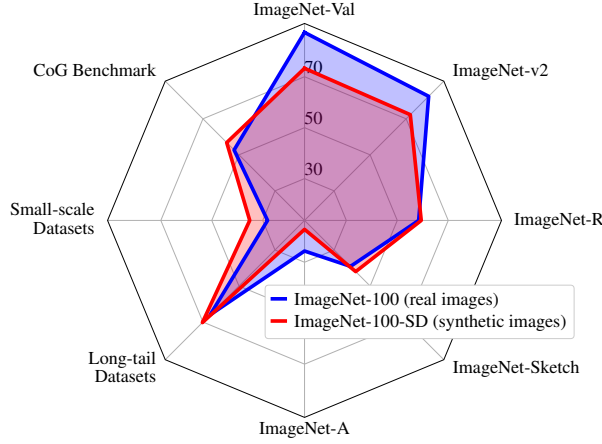
In Fig. 13, we show a few random images from each of the 100 classes in ImageNet-100, for three datasets: i) The real images from ImageNet-100, ii) synthetic images generated by a simple prompt, which is only composed of the name of the class, and iii) synthetic images generated with guidance scale equal to 2.0 and a prompt that enforces those classes to appear in diverse backgrounds to improve the diversity of generated images. From this exhaustive list, even with a few images per class, one can observe a number of issues around

the semantics, diversity and domain of those images.

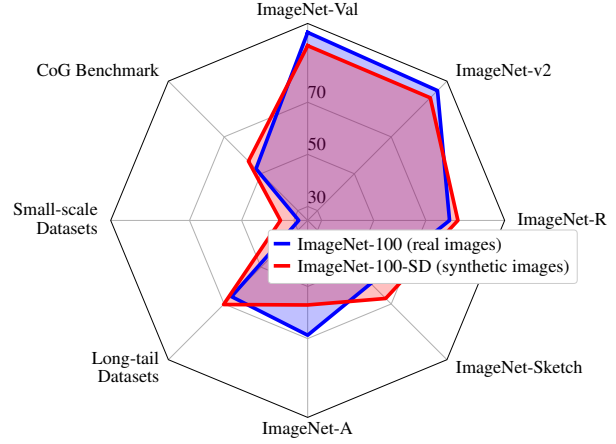
Showcasing domain and diversity issues. We also show extended results for three classes in order to illustrate issues related to the domain and diversity. Fig. 12 compares generated images between two fine-grained classes of crabs, while Fig. 11 shows many images from multiple different generated datasets for a single dog class. We discuss both figures in the next subsections.

D.1. Semantic errors

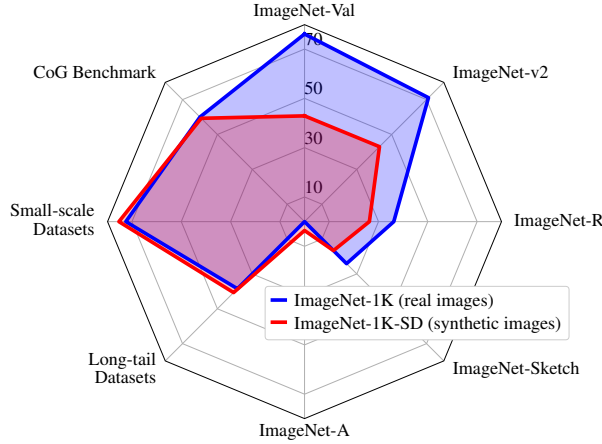
From closely inspecting the generated images we can see that there exists two classes for which the prompt $p_c = "c"$ produces images of the wrong semantics: For the classes “papillon” and “wing”, we see the generated images in the middle column of Fig. 13 to be wrong due to *polysemy* associated with the class names. What is more, although not fully visible from the small set of images we show here, we saw that semantics are partially wrong for at least the classes “green mamba”, “walking stick” and “iron”. For “green mamba”, although the synset refers to the snake species, there is a car model of the same name appearing in some of the generated images instead. For “walking stick”, the



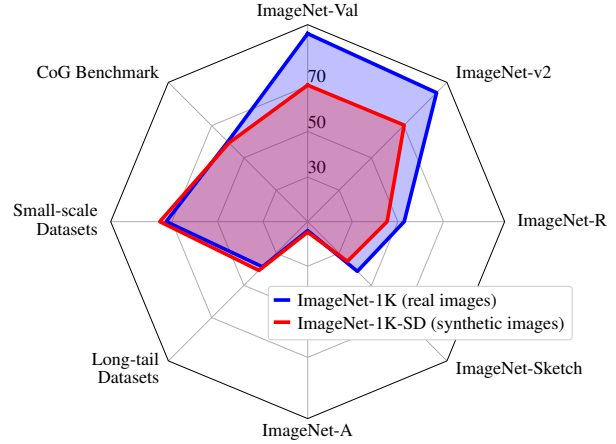
(a) Top-1 accuracy, training on **ImageNet-100**.



(b) Top-5 accuracy, training on **ImageNet-100** (top-1 for transfer tasks).



(c) Top-1 accuracy, training on **ImageNet-1K**.



(d) Top-5 accuracy, training on **ImageNet-1K** (top-1 for transfer tasks).

Figure 8. **Performance card of models** trained on either real or synthetic data for 100 classes of ImageNet-100 (Figs. 8a and 8b) and for all the 1000 classes of ImageNet-1K (Figs. 8c and 8d). In all figures, the **blue polygon** shows the performance of a model trained on the real images from ImageNet, and the **red polygon** depicts the performance of a model trained *only on synthetic data*, generated with Stable Diffusion [73] using $p_c = “c, h_c \text{ inside } b”$ as the prompt. In Figs. 8a and 8c and in Figs. 8b and 8d we report top-1 and top-5 accuracy over the ImageNet datasets (*i.e.*, ImageNet-Val/v2/R/A/Sketch), whereas, in all figures we report top-1 accuracy averaged over 8 transfer datasets. Note that Fig. 8d corresponds to Fig 1 of the main paper.

synset refers to the insect, while a subset of the generated images also contained walking sticks that are not insects.

As we discuss in the paper, appending the hypernym or definition of each synset seems to fix polysemy issues in many cases, including the ones mentioned above. However, we can see at least two cases where adding the hypernym in the prompt leads to worse results. According to WordNet [59], the hypernym for “shih-tzu” is “toy dog” something that results in dog-shaped toys in many of the generated images (see also Fig. 11). Another example is the class “boathouse”, where appending the parent class “shed” leads to sheds that are not inside a body of water.

D.2. NSFW content

Another issue that was not very prominent, but still visible, even in the case of generic animal and object categories present in ImageNet-100, was the fact that some of the generated images contained NSFW (Not Suitable For Work) content in the form of nudity. The open-source code for Stable Diffusion comes with a highly selective safety module, that discards generated images that might contain NSFW content.³ We disabled this module when generating images for the ImageNet synsets as we wanted to study the model

³<https://huggingface.co/CompVis/stable-diffusion-v1-4?text=Safety>

as-is first, and to understand the problem.

We thoroughly inspected all classes of ImageNet-100 and observed minor NSFW issues with two of the classes: 1) The basic prompt for the class “sarong” led to a few images that had partial nudity. This effect was exaggerated when adding the description of the concept that reads “a loose skirt consisting of brightly colored fabric wrapped around the body; worn by both women and men in the South Pacific”. It seems that words like “body” biases the image generation process towards more NSFW content. 2) Prompts for the class “ski mask” in combination with certain backgrounds from the Places dataset [102] also resulted in nudity. Overall, we want to emphasize that the Stable Diffusion models we tested were all highly susceptible to generate such content.

D.3. Misrepresentation of biodiversity

The degree of misrepresentation of biodiversity in the images generated from Stable Diffusion is very high. We partially showcase the issue in Fig. 12 where we show many generated images for two fine-grained classes, *i.e.*, “rock crab” and “fiddler crab”.

“Rock crab” is defined in WordNet as “crab of eastern coast of North America”, while the “fiddler crab” as a “burrowing crab of American coastal regions having one claw much enlarged in the male”. The fact that the male fiddler crab has one claw much larger is a prominent theme when it comes to the real ImageNet-100 images shown on the right side of Fig. 12a.

It does not take an expert ecologist to see that, although most of the generated images capture the coarser class “crab”, the visual differences between the two sets of images, *e.g.*, in Fig. 12b, are not focusing on the single enlarged claw for the fiddler crab case. What is more, the exhibited intra-class visual diversity, *i.e.*, crabs of different shapes and colors, seems to exceed a single species of crab.

This is just a single example, but from our inspection of many other fine-grained animal and fungi classes, we could see that this is not an isolated issue. On the contrary, it seems prominent across many fine-grained domains. One exception for the subset of ImageNet classes we delved into is dog breeds, possibly due to the sheer volume of dog images on the internet. It is however fair to say that the generated images highly misrepresent biodiversity.

It is worth noting that, as Luccioni and Rolnick discuss in their recent paper [52], the ImageNet dataset itself contains a number of issues when it comes to the annotations of fine-grained classes of wild animals. They found that “many of the classes are ill-defined or overlapping, and that 12% of the images are incorrectly labeled, with some classes having > 90% of images incorrect”. Although we did not conduct a similar experiment using experts, we expect similar statistics to be much higher for the images generated by Stable Diffusion.

D.4. Semantic issues arising with backgrounds

A common issue we observe when adding diverse backgrounds to class images is that a subset of the generated images do not really contain the object, and merely reflect the background scene. See for example the images in the first and last row, on the last column of Fig. 12c, and a few more spread in that figure, or the background samples for class “reel” in Fig. 13. This is to be expected given how a prompt like this is relying on the compositionality of the Stable Diffusion model.

What is really interesting is that in some cases the resulting images, although not containing an instance from the class, retains some of the object’s shape or texture in the background. See for example a pedestal-looking table in Fig. 12c for class “pedestal”, a pirate themed bedroom for class “pirate”, green shirts for “green mamba”, or the red-ish produce stand for “red fox”.

D.5. Issues with diversity

We observe issues with diversity for most of the classes when only the class name is used as the prompt, *e.g.*, in the middle set of results in Fig. 13. This is also visible for the crab classes in Fig. 12b, or the Shih-tzu class in Fig. 10b, Fig. 11a and Fig. 11b. We see that such issues are partially solved when lowering the guidance scale and relying less to the prompt, or using backgrounds (*e.g.*, the right-most set of images in Fig. 13). We expect more advanced prompt engineering to further increase diversity.

As expected, increasing diversity correlates with more semantic errors. We see that such issues appear far more frequently in the most diverse synthetic dataset, *i.e.*, as shown in the right-most set of images of Fig. 13.

D.6. Non-natural images

Even from the very small random sample of generated images shown in the figures of this paper, we see that there is a non-negligible percentage of the generated images that are non-natural. They can be illustrations, graphics images or even paintings. This is not necessarily undesirable and it can lead to models with higher robustness to related domain changes.

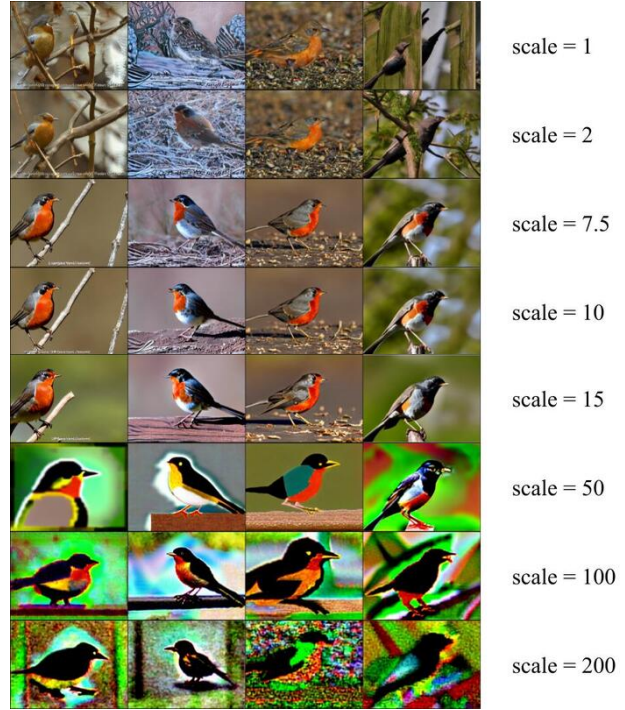
D.7. Varying the stable diffusion parameters

We identify two important parameters for Stable Diffusion, which affect the visual quality of generated images: The guidance scale and the number of diffusion steps. In Fig. 9 we show several examples where we vary one of these two parameters. More specifically, we generate images for the ImageNet synset n01558993 with class name “robin, American robin, *Turdus migratorius*”, for the simplest case where the prompt is just the class name. We fix the seed to 1947262 and vary either the guidance scale or the number of diffusion steps.

Guidance Scale. From Fig. 9a, we see that increasing the guidance scale coefficient over 10 starts giving hyper-realistic results. When the scale is under 2, we see that many details of the class are not really prominent.

Diffusion Steps. From Fig. 9b, we see that, although with 5 steps the generated images still contain a lot of noise, running 25-50 steps is enough for fully-formed, sharp images to emerge. Since this is a parameter that linearly impacts generation time, increasing the number of steps further than 50 seems excessive.

Output Resolution. The resolution that was used during training of the Stable Diffusion models was (512×512) .⁴ We notice that if one deviates from this training resolution, generated results get worse. We chose to simply switch the aspect ratio to the one for the average ImageNet image and keep the long dimension to 512.



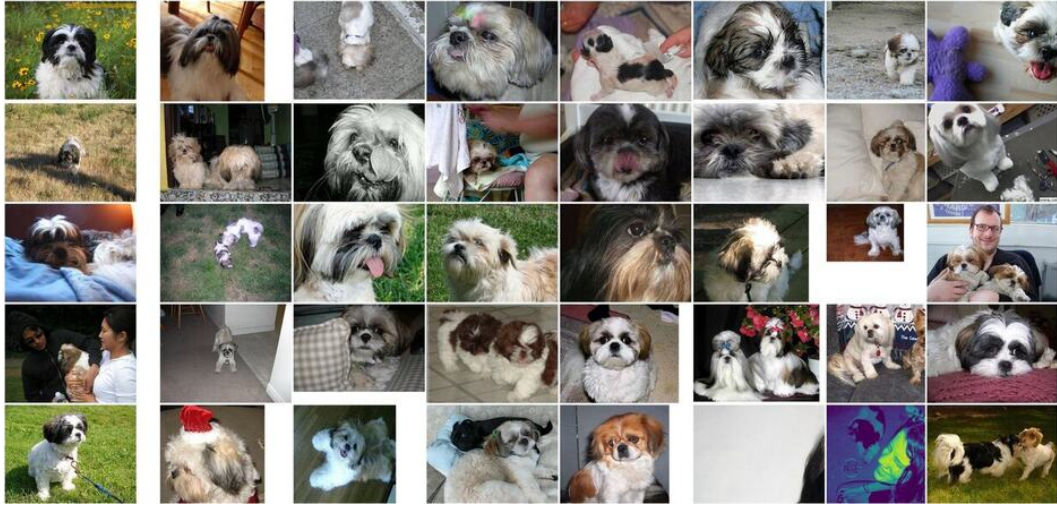
(a) Varying the guidance scale parameter (steps = 50)



(b) Varying the number of diffusion steps (scale = 7.5)

Figure 9. **Qualitative results as we change the guidance scale parameter and the number of diffusion steps during Stable Diffusion generation.** The seed is fixed to 1947262 and the prompt is “robin, American robin, *Turdus migratorius*”. Unless otherwise stated the scale (resp. steps) parameters are set to 7.5 (resp. 50).

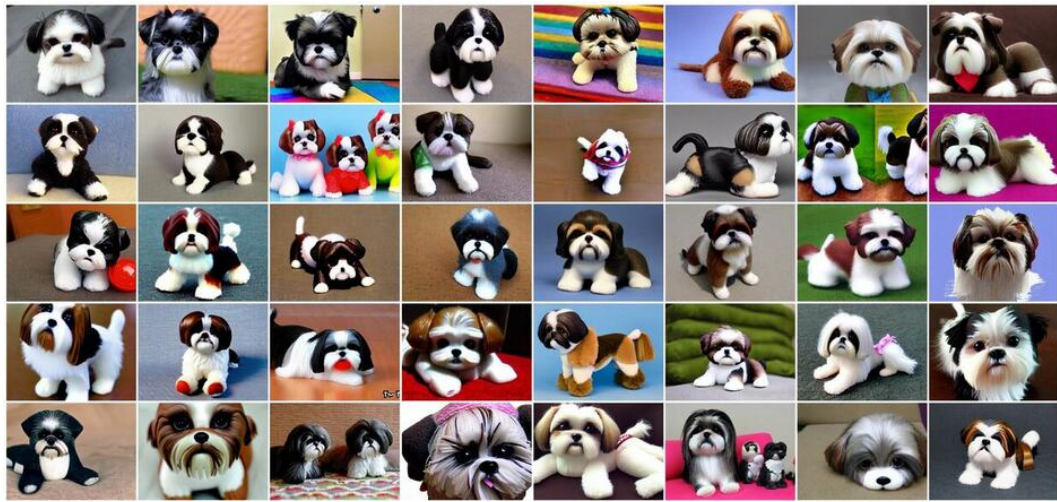
⁴<https://github.com/CompVis/stable-diffusion>



(a) Real images from ImageNet-1K for class “Shih-Tzu”



(b) Synthetic images with prompt $p_c = “c”$ for class “Shih-Tzu”

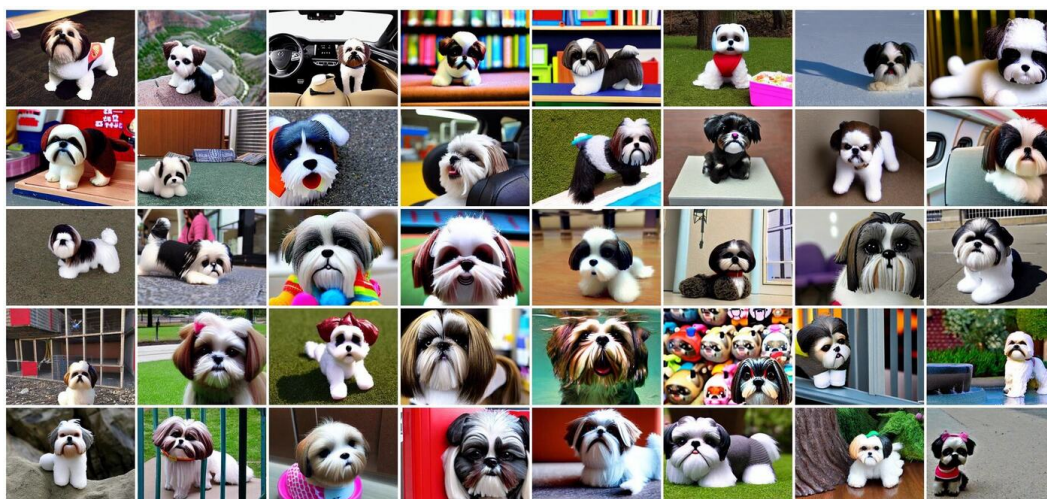


(c) Synthetic images with prompt $p_c = “c, h_c”$ for class “Shih-Tzu”

Figure 10. **Qualitative results for class “Shih-Tzu”** to illustrate domain and diversity issues. Guidance scale is equal to 7.5.

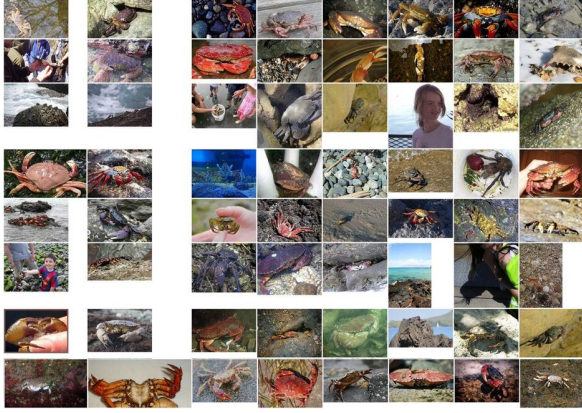


(a) (cont.) Synthetic images with prompt $p_c = "c, d_c"$ for class "Shih-Tzu"



(b) Synthetic images with prompt $p_c = "c, h_c \text{ inside } b"$

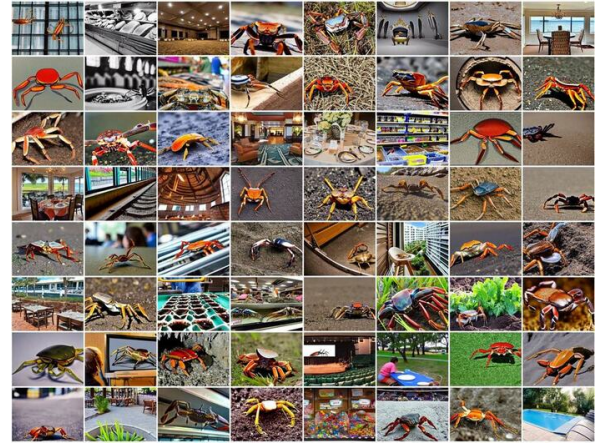
Figure 11. (cont.) Qualitative results for class "Shih-Tzu" to illustrate domain and diversity issues.



(a) Real images from ImageNet-1K for classes “Rock crab” (left) and “Fiddler crab” (right)



(b) Synthetic images with prompt $p_c = “c”$ for classes “Rock crab” (left) and “Fiddler crab” (right)



(c) Synthetic images with prompt $p_c = “c, h_c \text{ inside } b”$ for classes “Rock crab” (left) and “Fiddler crab” (right)

Figure 12. **Qualitative results for classes “Rock crab” (left) and “Fiddler crab” (right)**, to illustrate issues around fine-grained and domain specific semantics. Guidance scale is equal to 7.5.




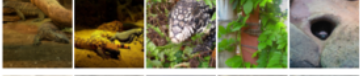


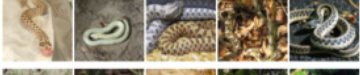
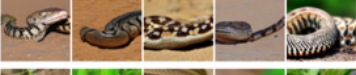

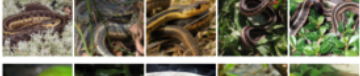
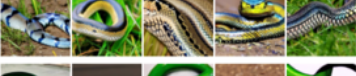
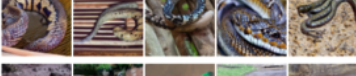
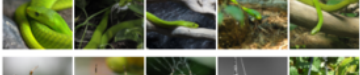
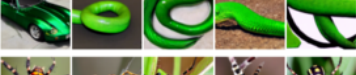
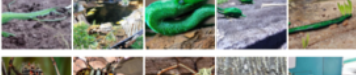
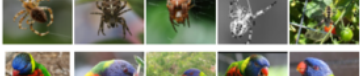
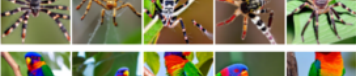
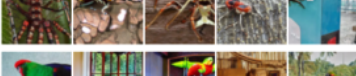



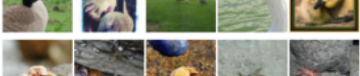
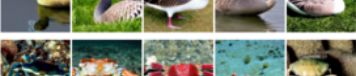
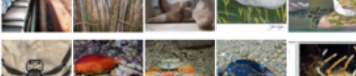
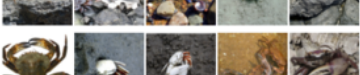
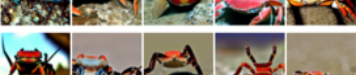

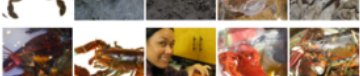
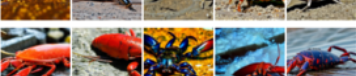


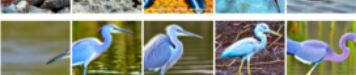

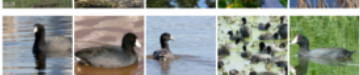

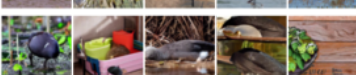

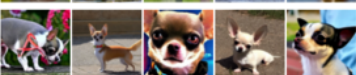





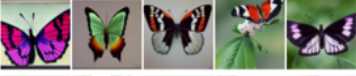
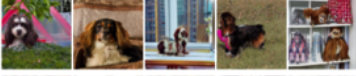

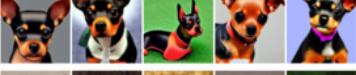
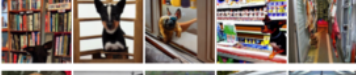

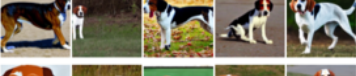


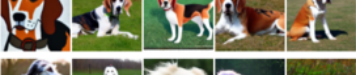




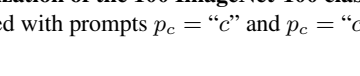
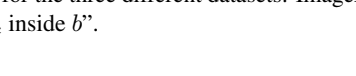
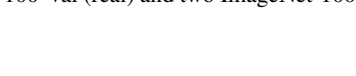
| Synset | real images | $p_c = "c"$ guidance scale 7.5 | $p_c = "c, h_c \text{ inside } b"$ guidance scale 2 |
|-------------------|---|--|---|
| robin |  |  |  |
| Gila monster |  |  |  |
| hognose snake |  |  |  |
| garter snake |  |  |  |
| green mamba |  |  |  |
| garden spider |  |  |  |
| lorikeet |  |  |  |
| goose |  |  |  |
| rock crab |  |  |  |
| fiddler crab |  |  |  |
| American lobster |  |  |  |
| little blue heron |  |  |  |
| American coot |  |  |  |
| Chihuahua |  |  |  |
| Shih-Tzu |  |  |  |
| papillon |  |  |  |
| toy terrier |  |  |  |
| Walker hound |  |  |  |
| English foxhound |  |  |  |
| borzoi |  |  |  |

Figure 13. **Visualization of the 100 ImageNet-100 classes** for the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c = "c"$ and $p_c = "c, h_c \text{ inside } b"$.

| Synset | real images | $p_c = "c"$ guidance scale 7.5 | $p_c = "c, h_c \text{ inside } b"$ guidance scale 2 |
|--------------------------------|-------------|-----------------------------------|--|
| Saluki | | | |
| American Staffordshire terrier | | | |
| Chesapeake Bay retriever | | | |
| vizsla | | | |
| kuvasz | | | |
| komondor | | | |
| Rottweiler | | | |
| Doberman | | | |
| boxer | | | |
| Great Dane | | | |
| standard poodle | | | |
| Mexican hairless | | | |
| coyote | | | |
| African hunting dog | | | |
| red fox | | | |
| tabby | | | |
| meerkat | | | |
| dung beetle | | | |
| walking stick | | | |
| leafhopper | | | |

Figure 14. (cont.) **Visualization of the 100 ImageNet-100 classes** for the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c = "c"$ and $p_c = "c, h_c \text{ inside } b"$.

| Synset | real images | $p_c = "c"$ guidance scale 7.5 | $p_c = "c, h_c \text{ inside } b"$ guidance scale 2 |
|-------------------|---|--|---|
| hare |  |  |  |
| wild boar |  |  |  |
| gibbon |  |  |  |
| langur |  |  |  |
| ambulance |  |  |  |
| bannister |  |  |  |
| bassinet |  |  |  |
| boathouse |  |  |  |
| bonnet |  |  |  |
| bottlecap |  |  |  |
| car wheel |  |  |  |
| chime |  |  |  |
| cinema |  |  |  |
| cocktail shaker |  |  |  |
| computer keyboard |  |  |  |
| Dutch oven |  |  |  |
| football helmet |  |  |  |
| gasmask |  |  |  |
| hard disc |  |  |  |
| harmonica |  |  |  |

Figure 15. (cont.) Visualization of the images for the 100 ImageNet-100 classes in the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c = "c"$ and $p_c = "c, h_c \text{ inside } b"$.

| Synset | real images | $p_c = "c"$ guidance scale 7.5 | $p_c = "c, h_c \text{ inside } b"$ guidance scale 2 |
|---------------|-------------|-----------------------------------|--|
| honeycomb | | | |
| iron | | | |
| jean | | | |
| lampshade | | | |
| laptop | | | |
| milk can | | | |
| mixing bowl | | | |
| modem | | | |
| moped | | | |
| mortarboard | | | |
| mousetrap | | | |
| obelisk | | | |
| park bench | | | |
| pedestal | | | |
| pickup | | | |
| pirate | | | |
| purse | | | |
| reel | | | |
| rocking chair | | | |
| rotisserie | | | |

Figure 16. (cont.) **Visualization of the 100 ImageNet-100 classes** for the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c = "c"$ and $p_c = "c, h_c \text{ inside } b"$.

| Synset | real images | $p_c = "c"$ guidance scale 7.5 | $p_c = "c, h_c \text{ inside } b"$ guidance scale 2 |
|-----------------|-------------|-----------------------------------|--|
| safety pin | | | |
| sarong | | | |
| ski mask | | | |
| slide rule | | | |
| stretcher | | | |
| theater curtain | | | |
| throne | | | |
| tile roof | | | |
| tripod | | | |
| tub | | | |
| vacuum | | | |
| window screen | | | |
| wing | | | |
| head cabbage | | | |
| cauliflower | | | |
| pineapple | | | |
| carbonara | | | |
| chocolate sauce | | | |
| gyromitra | | | |
| stinkhorn | | | |

Figure 17. (cont.) **Visualization of the 100 ImageNet-100 classes** for the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c = "c"$ and $p_c = "c, h_c \text{ inside } b"$.