Identifying Systematic Errors in Object Detectors with the SCROD Pipeline

Valentyn Boreiko[1,2], Matthias Hein[2], and Jan Hendrik Metzen[1]

[1]Bosch Center for Artificial Intelligence, Robert Bosch GmbH
[2]University of Tübingen

Abstract

The identification and removal of systematic errors in object detectors can be a prerequisite for their deployment in safety-critical applications like automated driving and robotics. Such systematic errors can for instance occur under very specific object poses (location, scale, orientation), object colors/textures, and backgrounds. Real images alone are unlikely to cover all relevant combinations. We overcome this limitation by generating synthetic images with fine-granular control. While generating synthetic images with physical simulators and hand-designed 3D assets allows fine-grained control over generated images, this approach is resource-intensive and has limited scalability. In contrast, using generative models is more scalable but less reliable in terms of fine-grained control. In this paper, we propose a novel framework that combines the strengths of both approaches. Our meticulously designed pipeline along with custom models enables us to generate street scenes with fine-grained control in a fully automated and scalable manner. Moreover, our framework introduces an evaluation setting that can serve as a benchmark for similar pipelines. This evaluation setting will contribute to advancing the field and promoting standardized testing procedures.

1. Introduction

Deep learning has significantly improved performance in many computer vision domains [21, 26]. However, models can display subpar performance on narrow but semantically coherent subgroups of the data. This can happen due to spurious features [33, 29], associations that a computer vision model has picked up when utilizing shortcut learning [9, 36, 19, 18]. While such shortcuts can lead to higher accuracy on the in-distribution data, they often fail on out-out-distribution data [42] and in the long-tail of the data distribution.

This non-homogeneous performance is problematic for applications such as automated driving, where models should work well on all subgroups of the data [1]. Therefore, recently there has been an increased interest in identifying subgroups of the data on which a computer vision model has subpar performance, which we call *systematic errors* [6, 13, 25, 35, 37]. While systematic errors have been studied for image classification [6, 13, 25, 37], multi-modal

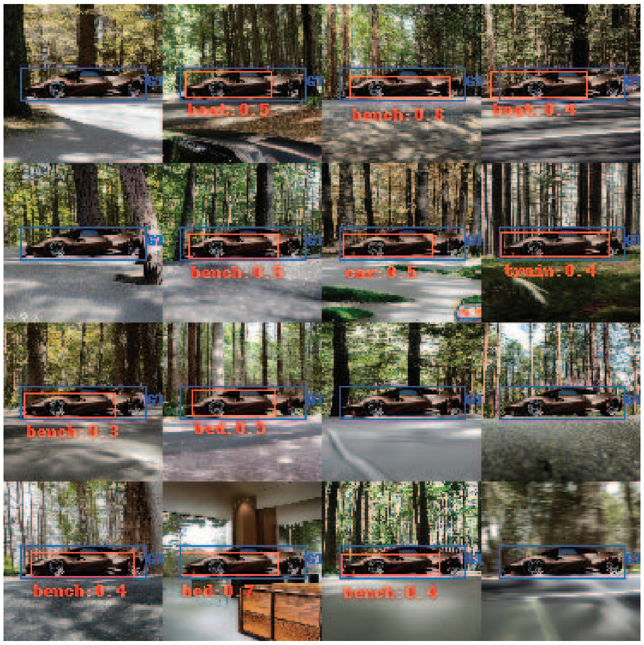


Figure 1: **Systematic errors detected with our SCROD pipeline into wrong classes *boat, bench, bed, and train* or False Negatives**. The prompt used is *"brown sports car is driving in forest, , in the morning, bright, idyllic, insanely detailed."*. Here, the image is downscaled by a factor of 5.5 before inputting it into the object detector, the rotation angle is −20°, and images are generated with 16 different seeds. Only in one case, it is detected as correct class **car**. The used object detector is *FasterRCNN_ResNet50_FPN* from torchvision [23].

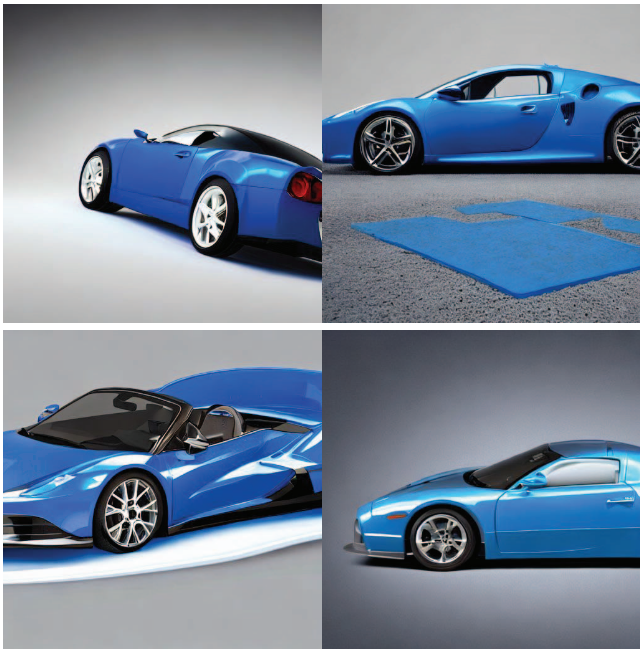


Figure 2: **Single generative models, such as a Stable Diffusion v1.5 (SD-v1.5) [32], do not allow to reliably generate images with fine-grained control,** ***while our SCROD pipeline can (see Fig. 5)***. In this figure, we show images generated by SD-v1.5 for 4 different seeds and the prompt *"blue sports car rotated by 50 degrees around the X axis from the side view on a grey background in the center of the image"*. In general, orientation is neither consistent nor correct, the object is cropped, and the color of the background and the main object is confused (upper right image, "attribute binding").

generative models [35] and image captioning [7], there is limited research for their discovery for object detectors, relying on a human-in-the-loop [7].

Real-world data is unlikely to cover all relevant subgroups in the tail of the data distribution. A promising alternative is to evaluate object detectors on synthetic data instead. Methods for synthesizing data can be roughly split into two categories: i) methods that rely on using physical simulators and hand-designed 3D assets [10, 31], which is labor-intensive and not scalable; ii) methods that rely on a single generative model, such as the Stable Diffusion v1.5 (SD-v1.5) [32], where geometric properties such object location, scale, or orientation are not easily controlled by textual prompts, as can be seen in the Fig. 2. Moreover, the well-known problem of *attribute binding* makes also object color/texture, and background control unreliable.

We address this challenge by combining several generative models (externally pre-trained as well as custom finetuned ones) in our novel pipeline (see Fig. 4). While we focus on the application of automated driving and thus on car objects, our method can be extended to other types of objects that are well represented in the distribution of the training data of the generative models we use in our pipeline. With this pipeline, we can identify systematic errors of object detectors (Fig. 1).

Our contributions are as follows:

- In Section 3, we propose a *novel pipeline for street scene synthesis*, allowing fine-grained control over attributes such as object location, scale, orientation, color, type, as well as scene background.
- We propose a custom model for *outpainting*, which is based on finetuning an inpainting model for street-scene outpainting, that we use in our street scene synthesis pipeline for background generation.
- In Section 4, we conduct an evaluation of the object detectors using our pipeline in two settings: one, where the background is a plain color and a second one where we show how our findings extend to more realistic (outpainted) backgrounds, where the background is generated conditioned on the text prompt. There we show some concrete systematic errors of the best object detectors from torchvision [23].

2. Related work

Controlled image generation. Controlled image generation can be achieved either by training generative models such as Stable Diffusion models from scratch with some guidance, such as inpainting guidance [32] or using methods for finetuning [40, 11, 27, 16, 38, 3], with guidance such as inpainting, Canny Edges, Depths Maps, Segmentation Maps, bounding boxes with target classes and many more. None of them can however offer a reliable fine-grained control over all the attributes at the same time that are useful for object detection: *location, scale, orientation, object color, object type, and background.*

Systematic error identification with subgroup annotations. This research direction mainly relies on creating reliable datasets, that can label as many attributes (or subgroups) as possible. Some examples are: DeepFashion2 [8] (attributes such as occlusions, segmentation, viewpoint, style, and category name are labeled), ImageNet-X [12] (attributes such as pose and background are labeled), **WEDGE** [24], **DAWN** [28], **nuScenes** [2] (attributes such as weather condition). While it is the most reliable way to test object detectors, it is not scalable as covering all possible combinations of relevant attributes is not feasible in general. Note moreover, that only the datasets in bold are related to the application of automated driving. Thus, this research offers limited utility for the systematic error identification of object detectors. Additionally, methods such as [10, 31] also offer subgroup annotations, but they rely on hand-made 3D assets or physical simulators.

Systematic error identification without subgroup annotations. Previous methods [6, 13, 25, 35, 37] have predominantly focused on the systematic errors of either classifiers or multi-modal models such as CLIP [30], and only a few, such as AdaVision [7], have done systematic error identification without subgroup annotations for object detection models. AdaVision, however, proposes a search over existing images with human-in-the-loop. It thus does not allow fine-grained control over the object attributes, as it is challenging to collect a dataset that encompasses all possible combinations of attributes, retrieve the relevant ones, and search through them automatically.

Novel View Synthesis. Control over the orientation of the object is important for automatically testing object detectors. Currently, this can be achieved either by using 3D assets and physical simulators as mentioned above or by using a generative model, such as Stable Diffusion, and fine-tuning it to predict novel views from a single view such as in Zero-1-to-3 [22]. This latter approach is promising for our pipeline and it can be further improved by fine-tuning on a bigger dataset [4].

3. Segment Control Rotate Outpaint Detect (SCROD) Pipeline

To identify systematic errors of object detectors, we require fine-grained control of the object properties such as pose. Using a single generative model, such as SD-v1.5, as has been done by Metzen et al. [25], does not allow such fine-grained control of details of the generated object to the degree which is necessary for the testing of object detectors, as can be seen in Fig. 2.

3.1. Workflow of SCROD

In contrast, we propose a multi-stage pipeline consisting of several generative models, where each model focuses on controlling specific properties of the generated objects and scenes. We focus on the properties such as *object type, color, location, scale, orientation, and background* of a given object of category "car". The whole pipeline is displayed step-by-step in Fig. 4. In the figure, we start with a real image of an object (in this case a side view of a black sports car). In the following, the runtime is reported for a batch size of 1 on an A100 GPU.

1. **Object type**: by using Segment Anything Model (SAM) [14] we segment objects of different car types automatically from the same fixed view (side view in our case). The inference time is 4 seconds.
2. **Object Color**: we extract Canny Edges of the segmented object and by using a ControlNet with Canny Edges conditioning [40], we then condition on the fine details of the automatically segmented object to create cars of different colors. Varying the color might introduce systematic errors for some object detectors as can be seen in the realistic setting in Fig. 7. The inference time is 20 seconds.
3. **Orientation**: by using a model such as Zero-1-to-3 [22], that was fine-tuned from SD on ObjaVerse [5] to generate views of the object, conditioned on the desired angles around X, Y, and Z axes, we can rotate an object around the X axis (pointing up). In our pipeline, we additionally use Stable Diffusion x4 upscaler (SD x4) [32] to improve the resolution of the generated images. The inference time is 2 seconds for the single view generation and 19 seconds for upscaling.
4. **Scale and Location**: we then downscale and change the location of the generated object by a different number of scaling factors, which does not require a separate model. Varying the scale and the location might introduce systematic errors for some object detectors as can be seen in the realistic setting in Fig. 6.
5. **Background**: In the experiments of this paper, we focus mainly on a plain color background, which does not require a separate model, and reduces the effects of background cues on object detector behavior. However, in case we want to outpaint a naturally looking background instead of using a plain color, we require an outpainting model. For this we use LoRA fine-tuning [11] of the SD-v1.5 inpainting model as described in Section 3.2. In the case of using the outpainting model, the inference time is 5 seconds.

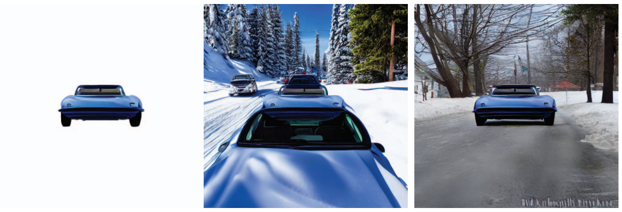
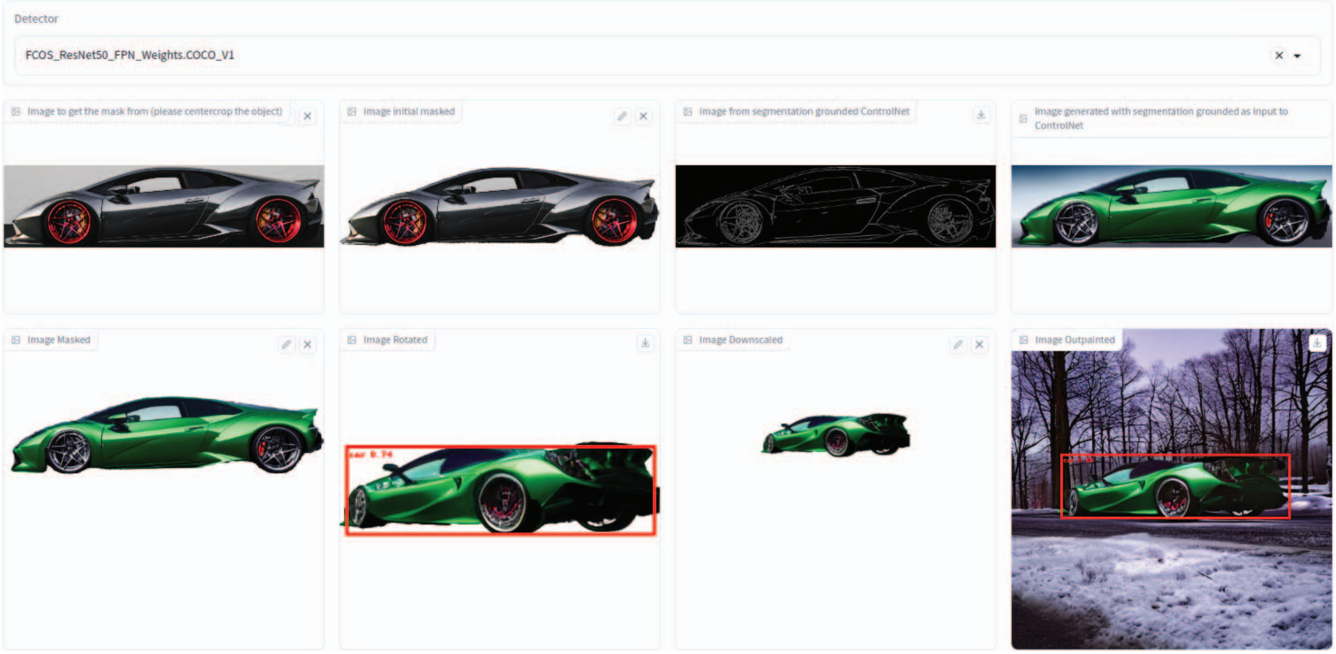


Figure 3: **Object outpainting**: *On the left* is the starting image of a synthetic car rotated by 90°. *In the middle* is the image outpainted with the SD-v1.5 inpainting model, where we observe over-generation: instead of preserving the boundary of the object, the outpainting continues hallucinating the object outside of the given boundary. *On the right* is the image outpainted with our proposed outpainting model, a LoRA fine-tuned [11] SD-v1.5 inpainting model (Section 3.2). Both outpaintings use the same seed and prompt *"sedan is driving on snowy street"*.

3.2. Outpainting model

To date, no strong models for masked-object outpainting exist and inpainting models tend to *over-generate*[1] the ob-

[1] We say an object is overgenerated if instead of preserving the boundary of the object during the outpainting continues hallucinating the object outside of the given boundary.



Detector

FCOS_ResNet50_FPN_Weights.COCO_V1

Image to get the mask from (please centercrop the object)
Image initial masked
Image from segmentation grounded ControlNet
Image generated with segmentation grounded as input to ControlNet
Image Masked
Image Rotated
car 0.74
Image Downscaled
Image Outpainted

Figure 4: **Depiction of our SCROD pipeline for synthesizing objects with certain attributes**. The workflow is as follows: *[Top, 1st from left]* We start with a real image of an object (in our experiments we always start with a side view of a car, but it can be chosen arbitrarily). *[Top, 2nd from left]* We segment the object itself and remove the background, using SAM [14]. *[Top, 3rd from left]* We extract Canny edge maps of the segmented object using a Canny edge detector. *[Top, 4th from left]* We invoke a ControlNet [40] to synthesize a new image, conditioned on the edge map and a text prompt (in the example "new green sports car"). The overall purpose of the top row is to control object attributes such as color (but also other types of textures or for instance, dirty-vs-clean could be controlled this way). *[Bottom,1st from left]* The generated synthetic image is then again segmented, and the background is removed using SAM. *[Bottom, 2nd from left]* The segmented object is then rotated using Zero-1-to-3 [22] by an externally specified rotation angle. Importantly, rotation does not require a 3D model of the object, just a segmented 2D view. Rotation can be around the axis left-right and top-bottom in principle (shown is only left-right). To improve image quality during the outpainting, we use Stable Diffusion x4 upscaler (SD x4) [32]. Object detectors under investigation can be evaluated on these intermediate results for a plain color background (which reduces dependencies on background cues). *[Bottom, 3rd from left]* The size and position of the object can be controlled by downsampling and translating the segmented object. *[Bottom, 4th from left]* The white background can finally be "outpainted" by conditioning a text-to-image model on the object and a text prompt like "sports car is driving on snowy street". We propose to do the outpainting using the fine-tuned model using LoRA [11] and starting from the SD-v1.5 inpainting model as described in Section 3. The object detector under investigation can then be tested on the resulting image such that for instance systematic errors on realistic object backgrounds can be identified.

ject as can be seen in Fig. 3 (please note how the boundary of the "car" object in middle image is not preserved and the object is transformed into a bigger one). We use LoRA fine-tuning [11] of the SD-v1.5 inpainting model on the joint dataset of COCO [20] and BDD100k [39], with the following training details: *number of training steps* is 15.000 , *LoRA rank* is 32, *batch size* is 8, and *learning rate* is 10^{-4}. The dataset $(\mathcal{X}, \mathcal{C}, \hat{\mathcal{X}})$ with $N = 19721$ samples has the structure as described below.

- **Labels during LoRA fine-tuning:** Ground truth original images $\hat{\mathcal{X}} = (\hat{x}_i)_{i=1}^N$.
- **Inputs during LoRA fine-tuning:** Ground truth segmented objects $\mathcal{X} = (x_i)_{i=1}^N$ provided from COCO and BDD100k (note that only 8.000 images from the whole BDD100k dataset have instance segmentations). During training, some objects in the segmentations masks are randomly dropped with probability 0.5 if other objects cover at least 10% of the area of the image. More precisely, $x_i = \hat{x}_i \cdot m_i$ for $i \in \{1, \ldots, N\}$, where m_i is a random binary mask obtained as described above. This is done to increase the diversity of the dataset by increasing the number of combinations of the masked objects shown on the image that are taken as an input. We consider objects from the following COCO classes as relevant to automated driving

applications: *car, person, truck, bus, traffic light, bicycle, motorcycle*; respectively from BDD100k: *rider, car, truck, bus, train, motorcycle, bicycle*.

- **Inputs during LoRA fine-tuning:** Captions $\mathcal{C} = (c_i)_{i=1}^N$, automatically generated using **BLIP-2**, OPT-2.7b model [15] $f_{\Theta,\mathrm{BLIP2}}$. That is, $c_i = f_{\Theta,\mathrm{BLIP2}}(\hat{x}_i)$.

Using this fine-tuned model, we can outpaint a realistic background, where the scene integrates the main object meaningfully, also generating shadows and reflections as can be seen in Fig. 4, 6 and 7.

4. Evaluation

We make use of the SCROD pipeline described in Section 3 to test the 3 best object detectors from torchvision [23] according to the Box MAP on COCO val2017 ($\mathrm{B_M}$):

- FasterRCNN_ResNet50_FPN_V2 (**FasterRCNN2**)[17]
- RetinaNet_ResNet50_FPN_V2 (**RetinaNet2**) [41]
- FCOS_ResNet50_FPN (**FCOS**) [34]

4.1. Color background

We start by varying the background colors and showing the systematic errors of object detectors when evaluated on the images generated with our pipeline and 11 different background colors over 16 seeds, when randomizing single view generation with Zero-1-to-3, for the generated object. In Tab. 1 we show some of the combinations of attributes for each of the object detectors with the highest error rate. An example of a systematic error and how minor changes to the underlying attributes can remove it is shown in Fig. 5.

By using our SCROD pipeline, we can find consistent and detector-specific systematic errors - that they occur predominantly for a single detector indicates that they are not caused by the generation process itself (for example, for the systematic error described in the first row of Tab. 1, images show cars and not airplanes, as is shown in Fig. 5). Also, we can highlight that systematic errors occur only in very specific subgroups and minor changes remove them. Lastly, the average error rates of detectors across all subgroups reported in the last row of Tab. 1 show that the object detectors can deal well with our synthetic images.

4.2. Realistic background

Moreover, we show systematic errors with a more complex (outpainted) background. Here we fix one seed for the generated object and vary the seed for the outpainted background. As can be seen from the reflections and shadows on images in Fig. 6 and 7, our proposed pipeline together with the custom model for outpainting allows us to generate scenes that look more natural and cannot be achieved by simply pasting an object into an existing background.

A more detailed analysis requires varying all possible single attributes and observing, which other combinations of the attributes can lead to the same systematic error, and which smallest changes of them can remove the systematic error. One example of such observation can be seen in Fig. 6 (c) and (d). From it, we can see that systematic errors are brittle and occur only under very controlled conditions.

By using our SCROD pipeline, we can easily identify such systematic errors of the object detector "FCOS" as a dependance on the scale of the object or on the angle of rotation of the object not only on the plain color background but also on a natural background, when outpainted using our LoRA fine-tuned model. This can be seen in Fig. 6. Here, we use 16 seeds when randomizing the outpainting.

1. In Fig. 6, (a) and (c), a purple SUV on the background outpainted using the prompt *"purple SUV is driving on beach, foggy, idyllic, lush detail."* is incorrectly detected as **bus** for all 16 seeds in (a). By changing the downscaling factor from 4.5 to 2.0, for 13 out of 16 seeds, objects are correctly detected as **car** (c).
2. In Fig. 6, (b) and (d), a purple coupe car on the background outpainted using the prompt *"purple coupe car is driving on street, foggy, in the morning, bright, stunning environment, sharp focus"* is incorrectly detected as **suitcase** for all 16 seeds in (b). By changing the rotation angle from $-80°$ to $-70°$, for 16 out of 16 seeds, objects are correctly detected as **car** (d).

Similarly, as displayed in Fig. 7, we can easily identify another systematic error of the object detector "RetinaNet2" as a dependence on the color of the object.

5. Conclusions

Summary. In this paper, we introduce a novel pipeline called SCROD, designed to automatically identify systematic errors in object detectors applied to synthetic street scenes. Our pipeline incorporates a custom outpainting model, enabling comprehensive error analysis in both plain color and natural backgrounds. We show the brittleness of even state-of-the-art object detectors from torchvision [23], highlighting the need for improved evaluation protocols.

Limitations. While SCROD provides extensive control over relevant attributes, the quality of control is still subject to the existing models' limitations. Additionally, our current implementation is object-centric and checks if a single object is detected correctly, potentially leaving out certain testing scenarios such as two objects occluding each other.

Outlook. Our SCROD pipeline is modular and building blocks such as ControlNet, Zero-1-to-3, or Outpainting can easily be replaced once better alternatives become available. So we expect the quality of our object detector testing pipeline SCROD will improve in the future. We leave it for future research to extend SCROD to handle multiple objects and occlusions within the scene. Additionally, developing more efficient non-brute-force search procedures is a promising direction.

Attributes					Error rates				
scale	angle	O	BG	type	YOLOv5n	FCOS	RetinaNet2	FasterRCNN2	YOLOv5x6
-	-90.0	black	blue	sports car	98% (airplane)	90% (mouse)	96% (mouse)	94% (kite)	83% (giraffe)
6.0	-50.0	-	grey	sports car	70% (airplane)	50% (kite)	4% (airplane)	**100%** (airplane)	0%
2.0	0.0	yellow	grey	sports car	94% (motorcycle)	31% (motorcycle)	**100%** (motorcycle)	0%	50% (motorcycle)
2.0	-90.0	pink	-	smart car	**100%** (truck)	**100%** (train)	55% (truck)	19% (kite)	**38%** (truck)
6.0	0.0	-	blue	sedan	**100%** (airplane)	8% (clock)	0%	0%	0%
Average Error Rate					59%	30%	17%	27%	7%

Table 1: **Combinations of attributes scale, angle, object color, background color, and car type (column "Attributes" on the left) that have the highest error rate across seeds and marginalized attributes (if any) in the corresponding group for 3 selected object detectors (column "Error rates" on the right)**. For each error rate, the class with the largest count of wrong predictions is displayed in parentheses for the object detector with the highest error rate. Here, we select the combinations of attributes that result in a significantly smaller error rate for the other 2 object detectors, to highlight that these systematic errors are detector-specific. When we marginalize one of the attributes, we put "-" in the respective subcolumn of the column **"Attributes"**. For each object detector, we report additionally the respective Box MAP on COCO val2017 (B_{M}). In the last row, we report average error rates per object detector across all subgroups. It shows that object detectors can deal well with our synthetic images.

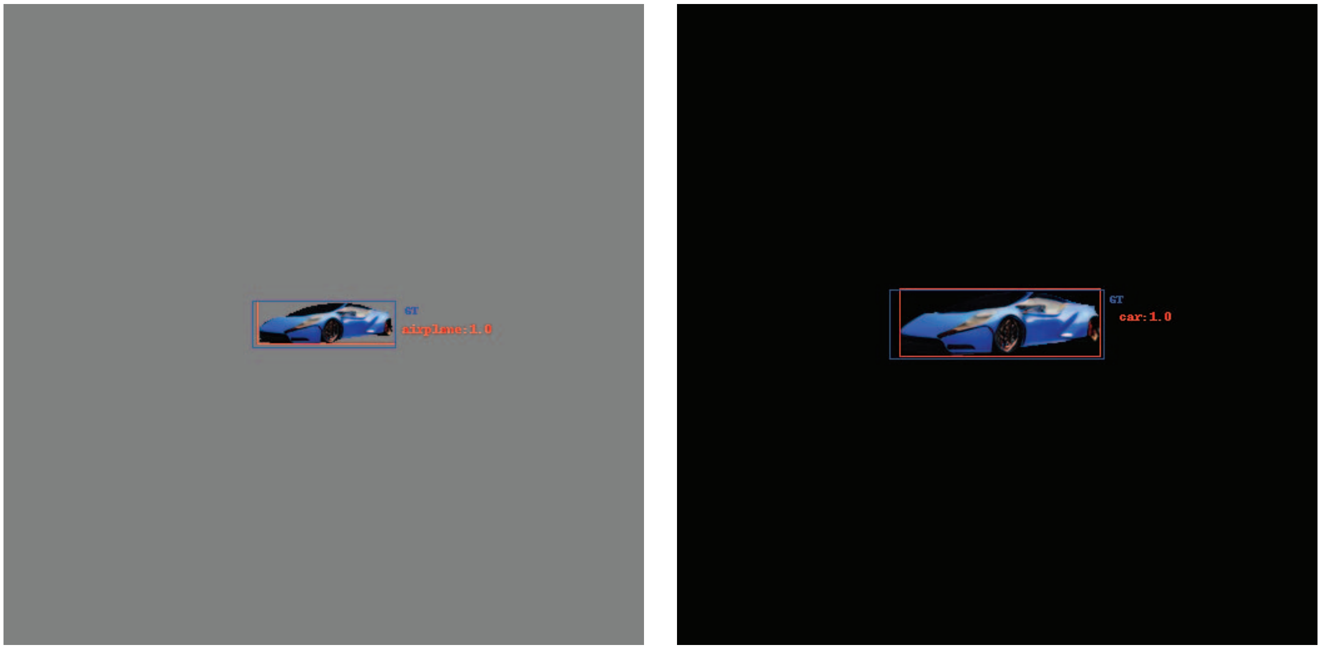
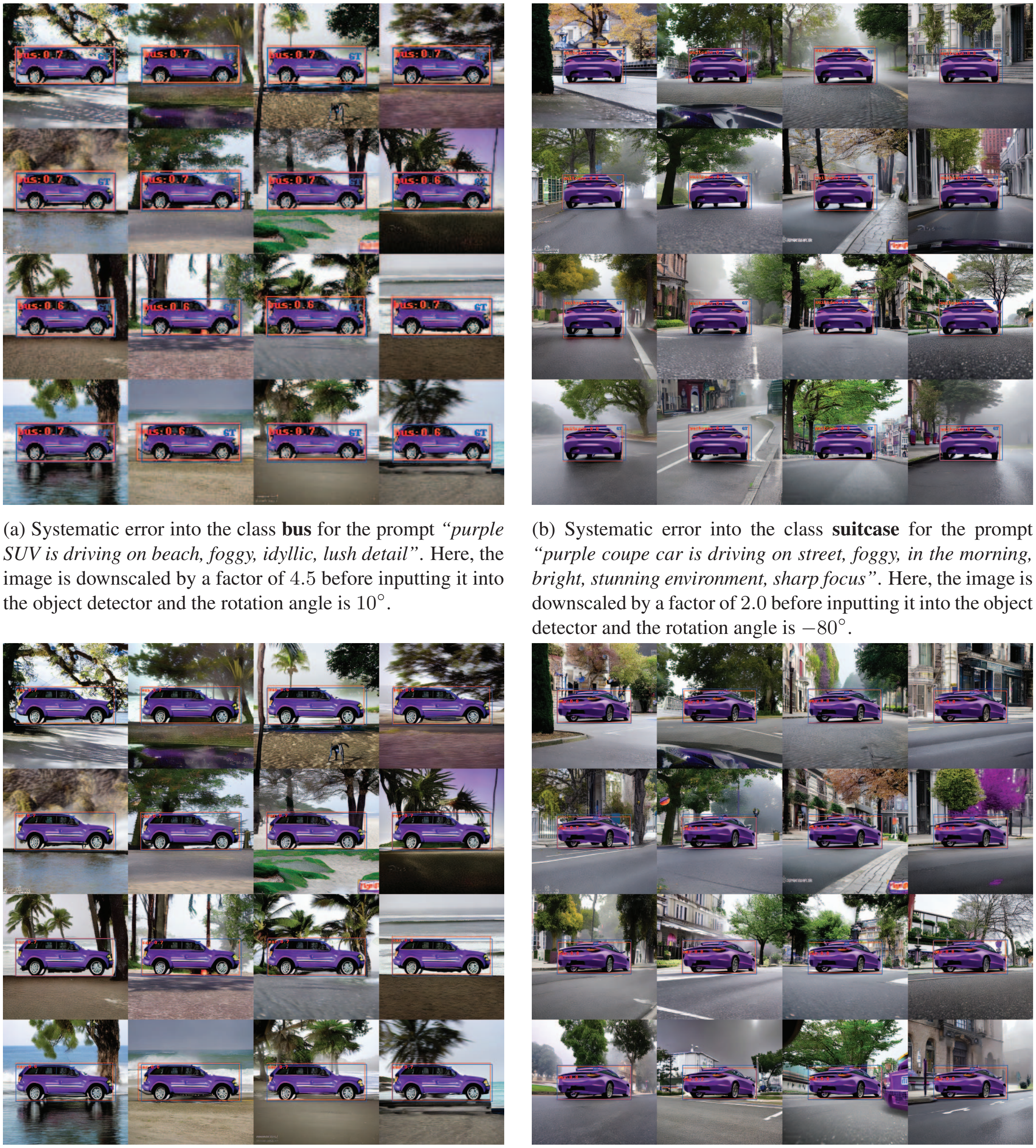


Figure 5: **Systematic error for the object detector "FasterRCNN2" motivated by the selected entry from the Tab. 1 (highlighted in blue)**. *On the left*, is the image, which is detected as the class **airplane**. This happens for the same attributes and same object detector across all 16 seeds when randomizing single view generation with Zero-1-to-3. Attributes used during the generation are the same as in the selected entry in the table. *On the right*, is the image for the same seed, but where the downscaling factor is decreased from 6.0 to 4.0 and the background color is changed to black.



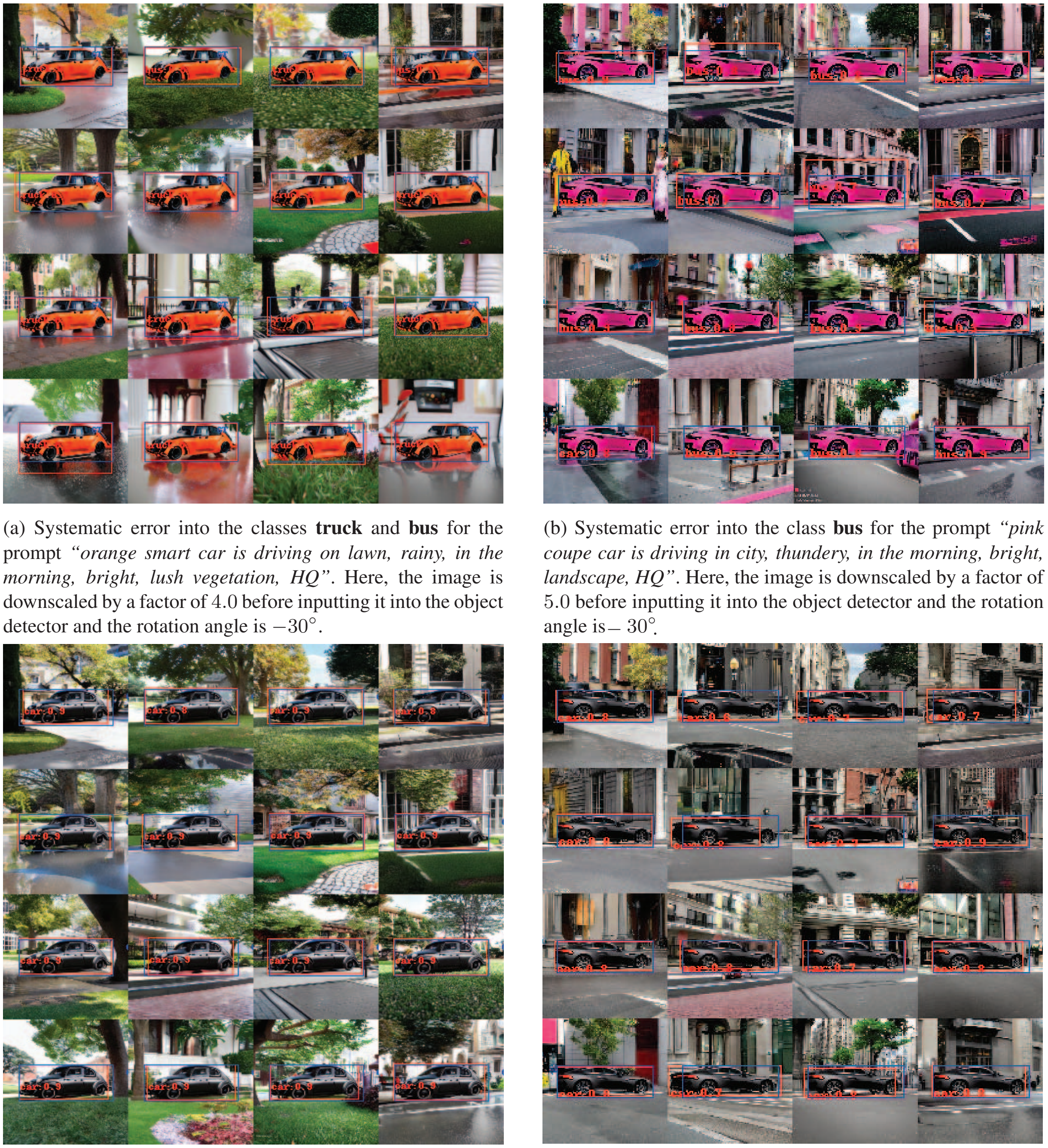
(a) Systematic error into the class **bus** for the prompt *"purple SUV is driving on beach, foggy, idyllic, lush detail"*. Here, the image is downscaled by a factor of 4.5 before inputting it into the object detector and the rotation angle is 10°.

(b) Systematic error into the class **suitcase** for the prompt *"purple coupe car is driving on street, foggy, in the morning, bright, stunning environment, sharp focus"*. Here, the image is downscaled by a factor of 2.0 before inputting it into the object detector and the rotation angle is $-80°$.

(c) Changing the prediction from the class **bus** to the correct class **car** for the prompt *"purple SUV is driving on beach, foggy, , idyllic, lush detail"* by changing the downscaling factor from 4.5 to 2.0 before inputting it into the object detector. Here the rotation angle is 10°.

(d) Changing the prediction from the class **suitcase** to the correct class **car** for the prompt *"purple coupe car is driving on street, foggy, in the morning, bright, stunning environment, sharp focus"* by changing the rotation angle from $-80°$ to $-70°$. Here, the image is downscaled by a factor of 2.0 before inputting it into the object detector.

Figure 6: **Subfigures (a) and (b) show detector-specific systematic errors for the object detector "FCOS"**. Subfigures (c) and (d) show that minor changes in object pose and scale result in correct predictions, which justifies that fine-granular control over image synthesis is required for identifying systematic errors. While all of the objects in (a) and (b) are detected as a wrong class with the object detector **"FCOS"** (error rate is 100%),**"RetinaNet2"** has error rates 13% (a), 25% (b), and **"FasterRCNN2"** - 0% in both cases.



(a) Systematic error into the classes **truck** and **bus** for the prompt *"orange smart car is driving on lawn, rainy, in the morning, bright, lush vegetation, HQ"*. Here, the image is downscaled by a factor of 4.0 before inputting it into the object detector and the rotation angle is $-30°$.

(b) Systematic error into the class **bus** for the prompt *"pink coupe car is driving in city, thundery, in the morning, bright, landscape, HQ"*. Here, the image is downscaled by a factor of 5.0 before inputting it into the object detector and the rotation angle is $-30°$.

(c) Changing the prediction from the classes **truck** and **bus** to the correct class **car** for the prompt *"black smart car is driving on lawn, rainy, in the morning, bright, lush vegetation, HQ"* by changing the color from "orange" to "black". Here, the image is downscaled by a factor of 4.0 before inputting it into the object detector and the rotation angle is $-30°$.

(d) Changing the prediction from the class **bus** to the correct class **car** for the prompt *"black coupe car is driving in city, thundery, in the morning, bright, landscape, HQ"* by changing the color from "pink" to "black". Here, the image is downscaled by a factor of 5.0 before inputting it into the object detector and the rotation angle is $-30°$.

Figure 7: **Subfigures (a) and (b) show detector-specific systematic errors for the object detectors "RetinaNet2" and "FCOS"**. Subfigures (c) and (d) show that minor changes in object color result in correct predictions, which justifies that fine-granular control over image synthesis is required for identifying systematic errors. While all of the objects in (a) and all but 1 in (b) are detected as a wrong class with the object detector **"RetinaNet2"** (error rate is 100% and 94% respectively), and all are detected as a wrong class with the object detector **"FCOS"**, detector **"FasterRCNN2"** has error rate 0% in both cases.

References

[1] Frederik Blank, Fabian Hüger, Michael Mock, and Thomas Stauner. Assurance methodology for in-vehicle AI. *ATZ worldwide*, 124:54–59, 07 2022.

[2] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020.

[3] Jiaxin Cheng, Xiao Liang, Xingjian Shi, Tong He, Tianjun Xiao, and Mu Li. Layoutdiffuse: Adapting foundational diffusion models for layout-to-image generation. *arXiv:2302.08908*, 2023.

[4] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-xl: A universe of 10m+ 3d objects. *arXiv:2307.05663*, 2023.

[5] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. *arXiv:2212.08051*, 2022.

[6] Sabri Eyuboglu, Maya Varma, Khaled Kamal Saab, Jean-Benoit Delbrouck, Christopher Lee-Messer, Jared Dunnmon, James Zou, and Christopher Re. Domino: Discovering Systematic Errors with Cross-Modal Embeddings. In *ICLR*, 2022.

[7] Irena Gao, Gabriel Ilharco, Scott Lundberg, and Marco Tulio Ribeiro. Adaptive testing of computer vision models. *arXiv:2212.02774*, 2022.

[8] Yuying Ge, Ruimao Zhang, Lingyun Wu, Xiaogang Wang, Xiaoou Tang, and Ping Luo. A versatile benchmark for detection, pose estimation, segmentation and re-identification of clothing images. In *CVPR*, 2019.

[9] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[10] Andrew Ilyas Sai Vemprala Logan Engstrom Vibhav Vineet Kai Xiao Pengchuan Zhang Shibani Santurkar Greg Yang Ashish Kapoor Aleksander Madry Guillaume Leclerc, Hadi Salman. 3db: A framework for debugging computer vision models. *arXiv:2106.03805*, 2021.

[11] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.

[12] Badr Youbi Idrissi, Diane Bouchacourt, Randall Balestriero, Ivan Evtimov, Caner Hazirbas, Nicolas Ballas, Pascal Vincent, Michal Drozdzal, David Lopez-Paz, and Mark Ibrahim. Imagenet-x: Understanding model mistakes with factor of variation annotations. *arXiv:2211.01866*, 2022.

[13] Saachi Jain, Hannah Lawrence, Ankur Moitra, and Aleksander Madry. Distilling Model Failures as Directions in Latent Space. *arXiv:2206.14754*, 2022.

[14] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.

[15] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023.

[16] Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. Gligen: Open-set grounded text-to-image generation. In *CVPR*, 2023.

[17] Yanghao Li, Saining Xie, Xinlei Chen, Piotr Dollar, Kaiming He, and Ross Girshick. Benchmarking detection transfer learning with vision transformers. *arXiv:2111.11429*, 2021.

[18] Zhiheng Li, Anthony Hoogs, and Chenliang Xu. Discover and mitigate unknown biases with debiasing alternate networks. In *ECCV*, 2022.

[19] Zhiheng Li and Chenliang Xu. Discover the unknown biased attribute of an image classifier. In *ICCV*, 2021.

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context. *arXiv:1405.0312*, 2015.

[21] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep Learning for Generic Object Detection: A Survey. *IJCV*, 128(2):261–318, Feb. 2020.

[22] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. *arXiv:2303.11328*, 2023.

[23] TorchVision maintainers and contributors. Torchvision: Pytorch's computer vision library. `https://github.com/pytorch/vision`.

[24] Aboli Marathe, Deva Ramanan, Rahee Walambe, and Ketan Kotecha. Wedge: A multi-weather autonomous driving dataset built from generative vision-language models. In *CVPR Workshops*, 2023.

[25] Jan Hendrik Metzen, Robin Hutmacher, N. Grace Hua, Valentyn Boreiko, and Dan Zhang. Identification of systematic errors of image classifiers on rare subgroups. In *ICCV*, 2023.

[26] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3523–3542, 2022.

[27] Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, Ying Shan, and Xiaohu Qie. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. *arXiv:2302.08453*, 2023.

[28] Mahmoud Hassaballah Mourad A. Kenk. Dawn: Vehicle detection in adverse weather nature dataset. `https://data.mendeley.com/datasets/766ygrbt8y/3`.

[29] Yannic Neuhaus, Maximilian Augustin, Valentyn Boreiko, and Matthias Hein. Spurious features everywhere – large-scale detection of harmful spurious features in imagenet. In *ICCV*, 2023.

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[31] Cinjon Resnick, Or Litany, Amlan Kar, Karsten Kreis, James Lucas, Kyunghyun Cho, and Sanja Fidler. Causal bert: Improving object detection by searching for challenging groups. In *ICCV Workshops*, 2021.

[32] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[33] Sahil Singla and Soheil Feizi. Salient imagenet: How to discover spurious features in deep learning? In *ICLR*, 2022.

[34] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *ICCV*, 2019.

[35] Shengbang Tong, Erik Jones, and Jacob Steinhardt. Mass-producing failures of multimodal systems with language models. *arXiv:2306.12105*, 2023.

[36] Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In *CVPR*, 2011.

[37] Olivia Wiles, Isabela Albuquerque, and Sven Gowal. Discovering bugs in vision models using off-the-shelf image generation and captioning. In *NeurIPS Workshops*, 2022.

[38] Han Xue, Zhiwu Huang, Qianru Sun, Li Song, and Wenjun Zhang. Freestyle layout-to-image synthesis. In *CVPR*, 2023.

[39] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *CVPR*, 2020.

[40] Lvmin Zhang and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *arXiv:2302.05543*, 2023.

[41] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z. Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *CVPR*, 2020.

[42] Chunting Zhou, Xuezhe Ma, Paul Michel, and Graham Neubig. Examining and combating spurious features under distribution shift. In *ICML*, 2021.