Efficient Exploration of Image Classifier Failures with Bayesian Optimization and Text-to-Image Models

Adrien Le Coz[1,2] Houssem Ouertatani[1,3] Stéphane Herbin[2] Faouzi Adjed[1]
[1]IRT SystemX, 91120 Palaiseau, France
[2]DTIS, ONERA, Université Paris Saclay F-91123 Palaiseau - France
[3]INRIA Lille - France
{adrien.le-coz, houssem.ouertatani, faouzi.adjed}@irt-systemx.fr stephane.herbin@onera.fr

Abstract

Image classifiers should be used with caution in the real world. Performance evaluated on a validation set may not reflect performance in the real world. In particular, classifiers may perform well for conditions that are frequently encountered during training, but poorly for other infrequent conditions. In this study, we hypothesize that recent advances in text-to-image generative models make them valuable for benchmarking computer vision models such as image classifiers: they can generate images conditioned by textual prompts that cause classifier failures, allowing failure conditions to be described with textual attributes. However, their generation cost becomes an issue when a large number of synthetic images need to be generated, which is the case when many different attribute combinations need to be tested. We propose an image classifier benchmarking method as an iterative process that alternates image generation, classifier evaluation, and attribute selection. This method efficiently explores the attributes that ultimately lead to poor behavior detection.

1. Introduction

In computer vision, deep learning models have achieved remarkable successes, consistently pushing the boundaries of what's possible in image classification [15], object detection [27], and many other applications. Despite these achievements, a persistent challenge remains: accurately discerning when the predictions made by these models can be trusted [2]. This is especially important for critical decision systems such as autonomous vehicles or medical imaging diagnostics. Even for less critical systems, errors have a cost that can be financial or reputational. The reliability of model predictions becomes particularly nebulous under conditions of data shift, inherent biases, and the presence of out-of-distribution (OOD) samples. Using pre-trained models can worsen those issues because the pre-training process and data might be unknown. It has been shown that deep neural networks often rely on spurious correlations for making predictions [13]. The conventional metric of a single accuracy number falls significantly short of comprehensively evaluating a model's performance. It is only a global evaluation of a given data distribution. New benchmarking tools are required.

Recently, there have been massive improvements in multimodal models, especially those combining textual and visual data like Text-to-Image generative models. These models have demonstrated an exceptional ability to understand and generate content that captures the nuanced interplay between text and images [26, 29]. This allows new ways of benchmarking image classifiers with generative models. Classifier performance can be studied in relation to the textual attributes of the data [22, 36, 37]. Despite their potential, however, the practical utility of these generative models is limited by the computationally intensive inference process of the underlying diffusion models. For example, in [37], testing whether the presence of a flower in an image causes the classifier to sometimes mistake flies for bees required hardware with 20×4 TPUs.

[22] developed a classifier evaluation process that starts with an Operational Design Domain [6] that textually describes the conditions the model is likely to encounter during use. It consists of many different combinations of attributes. To test a combination, they use synthetic data from a text-to-image model. They then identify which of these combinations lead to classifier errors. However, a major limitation is the combinatorial explosion: they need to limit the number of evaluated combinations. They suggest using combinatorial testing [24], but we found it far from optimal and not much better than random selection. In this paper, we are inspired by the principles of Bayesian Optimization (BO), a black-box global optimization method that is particularly well suited for functions with expen-

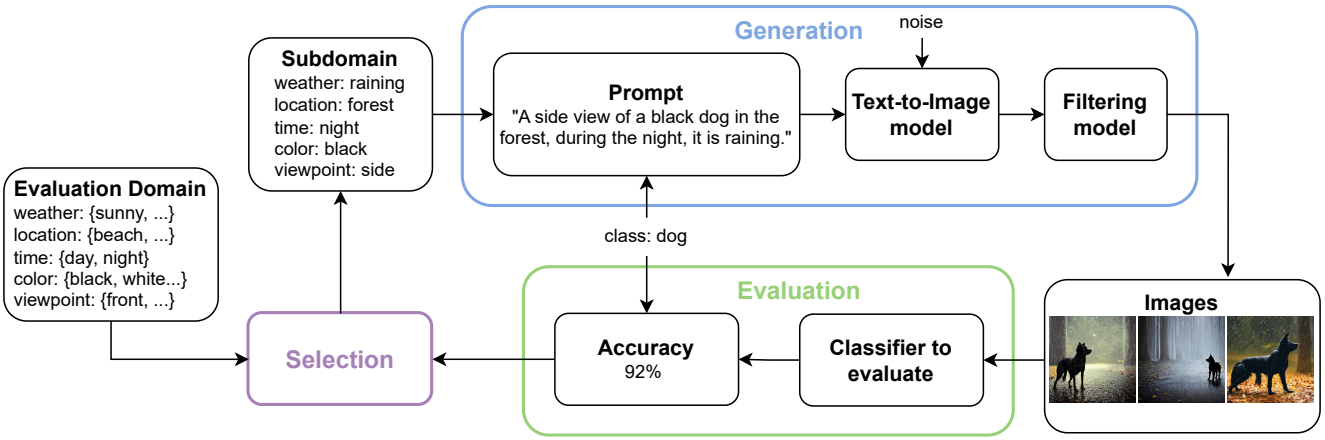


Figure 1. Illustration of our method that alternates generation, evaluation, and selection. The selection function selects the next subdomain to evaluate, based on the feedback of the previous subdomains evaluated. With the right choice of selection function, an efficient exploration of the evaluation domain is achieved.

sive evaluations. Among others, it has been successfully applied to Neural Architecture Search (NAS) [20]. We propose a novel approach to efficiently explore the semantic attributes of data that most significantly impact classification performance. By leveraging the insights gained from the multimodal models and addressing the limitations imposed by the computational demands of diffusion models, our approach seeks to enhance the reliability and interpretability of computer vision models. This paper details our methodology, which combines the strengths of Bayesian Optimization with the latest advancements in benchmarking computer vision with generative models. This offers a more efficient way to understand classification performance in relation to textual descriptions.

Our contributions are:

- We improve the efficiency of using Text-to-Image models to identify the textual attributes leading to classifier failure.
- We show that this approach significantly reduces the execution time while performing better than baselines.
- We demonstrate through some examples how our approach improves the understanding of classifier failures.

2. Related Work

Text-to-image generative models Diffusion models, an essential class of generative models, simulate the process of adding noise to data and then learning to reverse this process, enabling the generation of high-quality data samples. Introduced by [32], these models have paved the way for advancements in generative modeling by demonstrating how data distribution can be captured through denoising steps. The development of Denoising Diffusion Probabilistic Models (DDPMs) [16] marked a significant leap forward, refining training and sampling methods to produce high-fidelity images. Building upon these foundations, [8] introduced key improvements in efficiency and sample quality, leading to outperforming previously state-of-the-art generative models like GANs [14] and VAEs [21] in image quality and diversity. Textual conditioning allows for generating complex and diverse images by prompting them with text. Well-known Text-to-Image models include DALL-E 2 [26] and Imagen [29]. Stable Diffusion [28] emphasizes efficiency and scalability. It also makes high-quality text-to-image generation more accessible as it was published in open-source. While GANs can also be conditioned by text [30], the rapid improvements of diffusion models are hard to match. A main limitation of diffusion models is their inference time, requiring many denoising steps to generate an image. This is an important research avenue [8, 33].

Classifier failure discovery Discovering failures or bugs in image classification models has recently been studied more and more. One can use large labeled datasets and human verification to identify bugs [11]. To avoid these requirements, other approaches are based on generative models. In particular, leveraging recent Text-to-Image generative models allows linking textual attributes to classification performance. It is possible to identify bugs in a given classifier by generating many images and then clustering and captioning the ones leading to classification failure [37]. For instance, the presence of a flower in the images augments the chances of misclassification of flies into bees. However, the required computing resources are enormous. [36] personalizes the generation to a specific dataset to create distribution-shifted versions of the dataset. They can be used to test classification models' robustness to shifts. In our work, we can study combinations of shifts leading to failure, or in other words, corner cases. [22] identifies sub-

groups of data leading to degraded performance. Starting from an Operational Design Domain defined by domain experts and consisting of several semantic dimensions. An image classifier is tested on selected subgroups of this domain. We take inspiration from this work but derive a guided and efficient exploration of the attributes.

Bayesian optimization Bayesian optimization [12, 19] is often discussed in the context of Surrogate-Model Based Optimization (SMBO) [38]. The aim is to evaluate the costly objective function as few times as possible. To this end, an efficient model is used as its surrogate. BO typically relies on regression using Gaussian processes (GPs) in a process generally known as Kriging. Despite their ubiquity, thanks to many positive attributes, GPs have certain drawbacks. The most important one is the cubic complexity, making them inefficient as the observed data points increase. Their use is also contingent on selecting a kernel and possibly a distance function. It is however possible to effectively apply the general BO loop with alternative models, such as deep neural networks [31], as well as random forests [18] and Bayesian neural networks [12].

3. Method

We propose an efficient iterative process to explore the textual attributes leading to classifier failure. We first introduce the background concepts in subsection 3.1; our definitions for the evaluation domain and subdomains in subsection 3.2; the general pipeline to generate images for the subdomains in subsection 3.3; and our proposed guided exploration of attributes that matter in subsection 3.4.

3.1. Background

Image classifier To demonstrate our approach without using considerable computing power, we tackle a simplified task: binary classification of images containing dogs. We construct a dog classifier from a classifier pre-trained on ImageNet [7], a dataset that contains images of animals or everyday objects. Out of the 1000 classes, 119 are different dog breeds. We sum the classifier probabilities of these classes to get the *dog* probability and sum the rest to get the *not-dog* probability.

Text-to-image generative models We use diffusion models as a method for generating images from textual descriptions. They are characterized by their ability to produce high-quality images through a process of denoising. The core mechanism involves a forward diffusion process that incrementally adds noise to an image until it becomes indistinguishable from Gaussian noise. The reverse process, iteratively reconstructing the image from noise, is learned during training. For Text-to-Image models, the reverse process is conditioned on textual descriptions of the images. Specifically, on embeddings derived from a pre-trained language model to ensure the generated images align with the provided textual descriptions. This conditioning is usually integrated with cross-attention [35].

Algorithm 1 Exploration of image classifier failures

Input:
D_{to_eval} the evaluation domain
S_{to_eval} the list of subdomains to evaluate
f: the classifier
g: the generative model
h: the selection function
n: the number of allowed evaluations
$S_{eval} = \emptyset$ the dataset of subdomains evaluations
$s_0 \in S_{to\_eval}$: the initial selected subdomain
Explore subdomains:
for $i = 0$ **to** n **do**
  Generation
  build prompt p_i from selected subdomain s_i
  $\hat{x_i} \leftarrow g(p_i)$ ▷ generate and filter images from prompt
  Evaluation
  $\hat{y_i} \leftarrow \arg\max f(\hat{x_i})$ ▷ compute predicted classes
  $a_i \leftarrow acc(y, \hat{y})$ ▷ compute classifier accuracy
  Selection
  $S_{eval} \leftarrow S_{eval} \cup \{(s_i, a_i)\}$ ▷ add result to dataset
  $S_{to\_eval} \leftarrow S_{to\_eval} \setminus s_i$ ▷ remove from list
  $s_{i+1} = h(S_{eval}, S_{to\_eval})$ ▷ update h and select next
end for

3.2. Define the evaluation domain and subdomains

We call *evaluation domain* the ensemble of deployment environment conditions to evaluate. The conditions are described by textual attributes, each containing a finite number of values. They can be categorical or continuous, but we focus on categorical attributes in this work. The domain comprises all the possible attribute value combinations, which we call *subdomains*. The number of subdomains grows exponentially with the number of attributes considered.

As a starting point, we need to define the textual attributes to explore. Expert knowledge is thus required. As we study image classification of natural images of dogs, we define the following attributes and associated values in brackets: weather [sunny, cloudy, raining, snowing], location [at the beach, in the forest, in the city, inside a house, in a garden, in the desert, in the mountains], time [day, night], color [white, black, brown, beige, gray, red, green, blue], and viewpoint [front, side, rear]. Some combinations are not valid and must be removed.

(a) "A side view of a brown dog in the city, during the day, it is sunny."
(b) "A front view of a white dog at the beach, during the day, it is sunny."
(c) "A rear view of a black dog in the desert, during the day, it is foggy."
(d) "A front view of a red dog in the city, during the day, it is snowing."

(e) "A front view of a white dog at the beach, during the day, it is sunny."
(f) "A side view of a beige dog in a garden, during the day, it is raining."
(g) "A rear view of a blue dog in the mountains, during the night, it is foggy."
(h) "A front view of a beige dog inside a house, during the day, it is sunny."

Figure 2. Samples of generated images with their associated prompt. Images on the top row are classified as dogs, while those at the bottom are not. Note that some biases of the generative model appear: sunglasses at the beach and an umbrella when raining.

3.3. Generate data conditioned by attributes

Prompt The first step is to create a textual prompt corresponding to one subdomain attribute. We use a prompt template to fill with the attributes: "A {viewpoint} view of a {color} dog {location}, during the {time}, it is {weather}.".

Generate A Text-to-Image model can then generate images conditioned by the textual prompt. The generation is not deterministic: the starting noisy image is random, and noise is applied to each step of the reverse diffusion process. This means that one textual conditioning leads to a variety of aligned images.

Filter The generation is not perfect, and sometimes the synthetic image does not align well with the textual prompt input. We derive a filtering process that follows the generation to limit this issue. We use CLIP [25] as a zero-shot subdomain classifier. We have a finite number of subdomains, and each of them is defined as a textual prompt. We thus compute the cosine similarity between a generated image and all subdomains prompts to obtain logits. Applying the softmax function to the logits, we get predicted probabilities that the image corresponds to each subdomain. If the prompt with the maximum probability is indeed the prompt used to generate the image, we consider the image correct otherwise it is filtered out.

Subdomain index	Viewpoint	Color	Time	Location	Weather	Classifier accuracy
0	side	white	day	at the beach	sunny	0.98
1	side	white	day	at the beach	snowing	0.94
2	side	white	day	at the beach	raining	0.86
...	...	...	...	...	...	...
1031	rear	blue	night	in the mountains	foggy	0.66

Table 1. Reference evaluation data. The generation and evaluation steps were pre-computed, and the results were saved in a table. A table look-up replaces these costly steps to compare different selection functions quickly. This also removes the variance in the process.

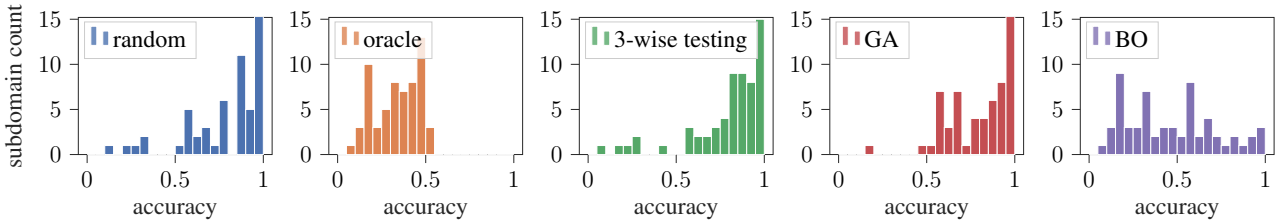


Figure 3. 3-wise testing selects 61 subdomains to evaluate. Most of them are high-accuracy. We compare that to the other methods when allowed to explore 61 subdomains. GA and Bayesian optimization identify much more low-accuracy subdomains.

3.4. Guided exploration of attributes that matter

Because generating data conditioned by the attributes described above is time-consuming, we propose an efficient exploration of the critical attributes. An iterative process alternates the generation of images for a subdomain, evaluates the classifier on the subdomain, and selects the next subdomain to evaluate based on this feedback. The process is described schematically in Figure 1 and more formally in Algorithm 1. We propose several selection functions below.

Genetic algorithm (GA) This is a performant optimization method based on natural selection [17]. A population of solutions is evaluated. The top performers are preserved, and a crossover operation generates *children* solutions from pairs of *parents*. This new generation of solutions undergoes mutations with a small probability, adding diversity.

Bayesian optimization (BO) Our method to efficiently explore the space of subdomains involves the same core loop at the center of Bayesian optimization, relying on a predictive model to guide the search towards the critical subdomains.

1. Selection: choose the next subdomain to evaluate using the model
2. Observation: evaluate the subdomain
3. Model update: add the new observation to the dataset

The selection policy generally means selecting the point which maximizes an acquisition function. Many acquisition functions exist in the literature, and each presents a different trade-off between exploration and exploitation. The selection policy we use is inspired by Expected Improvement [23], a widely used and generally effective acquisition function. Using the model's estimation of each subdomain's quality, we select the subdomain with the highest potential improvement over the current best subdomain.

4. Experiments

We conduct experiments evaluating the different aspects of our approach. We first provide information on our experimental setting in subsection 4.1; we provide details on the reference data generation in subsection 4.2; we compare different selection functions, including some baselines, in subsection 4.3; and we display qualitative results of classifier evaluation in 4.4.

4.1. Prerequisites

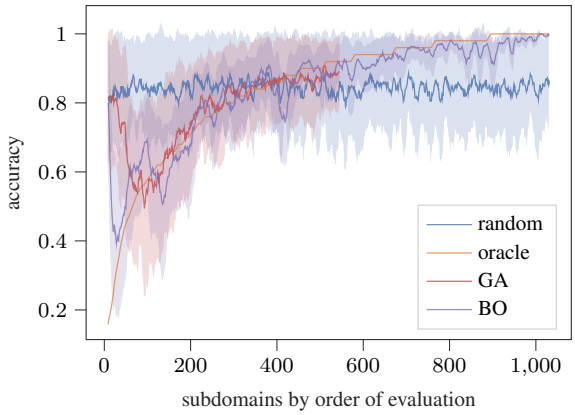
Classifier We study a classifier with the ViT-B/16 [9] architecture. Weights are from torchvision, following a pretraining on the ImageNet dataset. The binary classifier's accuracy on ImageNet validation data is more than 99%. We want to assess its performance on data that is more diverse than in the original dataset to see if it can generalize well.

Subdomains The number of possible attribute combinations is 1 class (dog) $\times$ 4 weathers $\times$ 7 locations $\times$ 2 time periods $\times$ 8 colors $\times$ 3 viewpoints = 1344. However, some of the combinations are impossible (e.g., "during the night, it is sunny" or "in a house, it is snowing"). After filtering those, 1032 combinations remain, forming all the possible subdomains to evaluate.

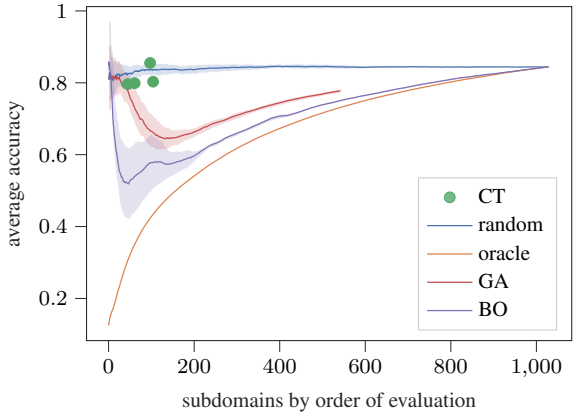
Generative model We use Stability AI's implementation of Stable Diffusion 2.1 as a text-to-image generative model. Its architecture is based on Latent Diffusion Models [28], and text conditioning uses a fixed pre-trained text encoder based on CLIP ViT/H. Generated images have a 512×512 resolution, but we resize them into 256×256 to save disk space. Resizing images at a lower resolution is part of the classifier data preprocessing anyway. We treat this model as a black box transforming textual input prompts into diverse corresponding images.

Filtering model Because the generation is imperfect, we need to filter out generated images that do not align well with the textual input prompt. We use a subdomain classifier that classifies generated images into one of the subdomains. This classifier is a pre-trained CLIP ViT-L/14 adapted as a zero-shot classifier.

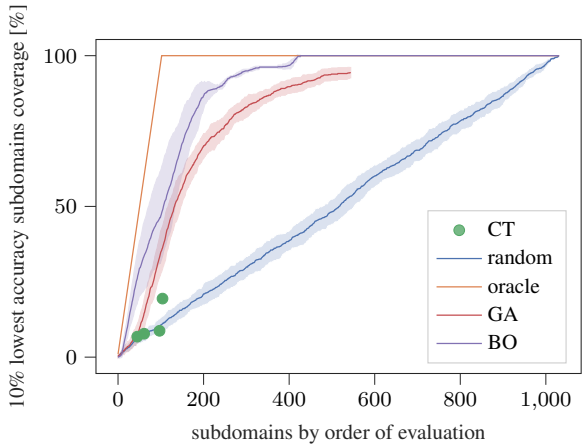
Baselines For comparison, we include some methods of selecting the subdomains to evaluate as baselines.



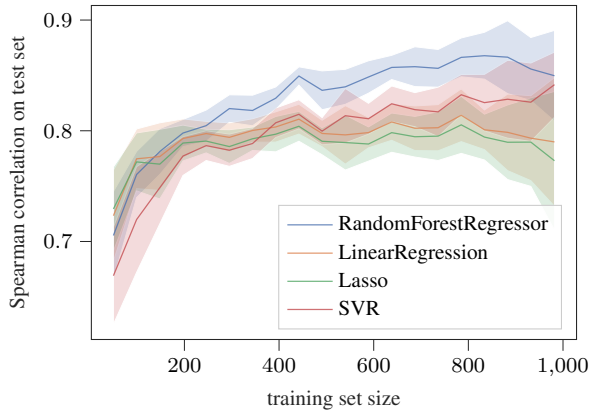
(a) Evolution of the accuracy of selected subdomains during the exploration (lower is better). We used a moving average with a window size of 10 to improve clarity. GA and the BO quickly select low-accuracy subdomains until only higher-accuracy subdomains remain.



(b) Evolution of the average accuracy on subdomains already evaluated during the exploration (lower is better). All methods converge to the global accuracy. Combinatorial testing is not much better than random selection, compared to the GA and BO.



(c) Evolution of the 10% lowest accuracy subdomains coverage (higher is better). We identified the 10% (103) subdomains with the lowest accuracies and computed what proportion of them is covered by the subdomains selected during the exploration. The BO finds all of them after evaluating $\approx$ 300.



(d) Spearman's rank correlation coefficient for different predictors and training set sizes. It measures the strength and direction of the monotonic relationship between two ranked variables, here the predicted and test accuracies. A value close to 1 means the relationship between the two variables is monotonic. Except for SVR, all predictors perform similarly well. Lasso is the best method for small training sizes.

Figure 4. Different metrics to compare the quality of the subdomain selection when iterating on the loop generation, evaluation, and selection. In general, combinatorial testing is not much better than random selection, and it only gives a few options for the number of subdomains selected. GA and BO are much more efficient and can explore any given number of subdomains according to the computation time available. Note that the x-axis of 4a, 4b, and 4c could be replaced by GPU.hours going from 0 to $\approx$ 200 as mentioned in Subsection 4.2. All plots are averages over 10 seeds and the standard deviations are shown.

- The *random selection* simply randomly picks a subdomain to test in the list of the remaining ones.
- The *oracle* knows all the subdomain's accuracies in advance, and it chooses the subdomains by order of increasing accuracy. This is the best way to select the subdomains, but also the most costly as it requires knowing all the subdomains' performances.
- *Combinatorial Testing (CT)* [24] aims to test a limited number of combinations that cover well the search space. In particular, we use n-wise testing from the library allpairspy [1]. We vary n from 2 (pairwise testing) to 5 (because we have 5 attributes). This approach was used by [22].

Methods details We test two different approaches:

- *Genetic algorithm (GA)* We use a population size of 20 and the library pymoo [3].
- *Bayesian optimization (BO)* The predictor takes a one-hot embedding of the subdomain attributes as input and pre-

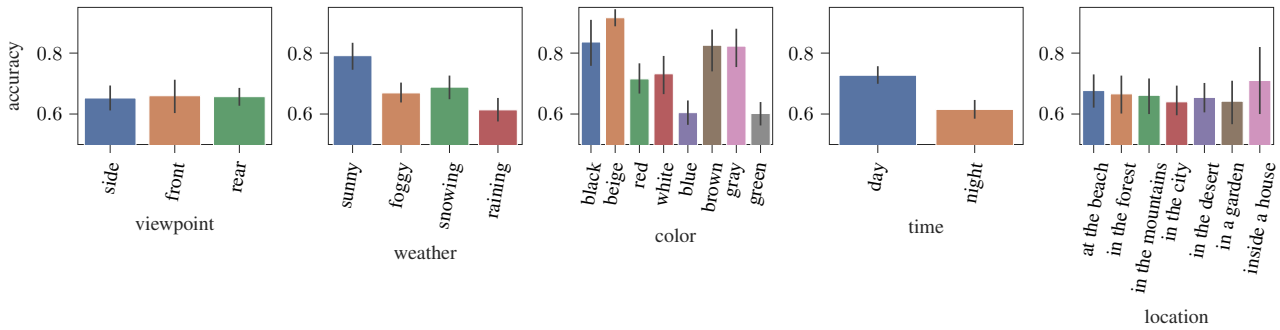


Figure 5. Average accuracies for each value of each attribute. The 95% confidence interval is also shown.

dicts the accuracy. We tested Random Forest Regressors (RFR) [4], Linear Regression (LR), Lasso [34], and Support Vector Regression (SVR) [10] using scikit-learn [5]. We start with a pre-training on 10 random subdomains to lower the variability of each run.

Metrics We study the evolution of selected subdomain accuracies, average accuracy of selected subdomains, and coverage of the 10% lowest accuracies subdomains. We also show a histogram of the subdomain accuracies for a fixed number of explored subdomains. We use the Spearman rank correlation to evaluate the quality of the predictors.

4.2. Evaluating all subdomains for reference

To validate our approach, we evaluate the performance of all subdomains and save the results as shown in Table 1. Because all evaluation results are pre-computed, benchmarking the different selection functions is done by replacing the generation and evaluation parts with a simple table lookup. This allows us to compare different selection functions quickly. This validation ensures that subsequent work can use our findings to reduce the number of evaluations. We generated 50 valid images for each of the 1032 subdomains. It took approximately 200 hours to generate all images on one NVIDIA V100 GPU. Sometimes, hundreds of images had to be generated to obtain 50 valid ones after filtering. The expected evaluation time of one subdomain is 12 minutes, or 1 hour for 5 subdomains. The results below show the number of subdomains explored as the x-axis. Still, we could have used an estimated computing time by using the value of around 12 minutes per subdomain evaluated.

Figure 2 shows samples of generative images with their input prompt. While not perfect depictions of dogs, they are close enough to benchmark the classifier. Some images clearly depict dogs, yet the classifier fails to identify them. This highlights some of its limits.

4.3. Benchmarking the selection functions

The main goal of selection functions is to identify subdomains with low accuracy quickly. To measure this, we show the evolution of different metrics during the exploration in Figure 4. The main conclusion is that combinatorial testing (n-wise testing with $n \in \{2, 3, 4, 5\}$) is not much better than random selection. Also, it has the disadvantage of restricting the number of subdomains selected: we cannot tune this number. GA is much better, and BO is even better. BO can successfully identify all the 10% most critical subdomains (with lowest accuracies) after evaluating $\approx 40\%$ of all subdomains. This also proves that subdomain performance can be precisely inferred from the domain attributes. This means that classifier failures can be explained from the attributes, providing interesting insights into the classifier decision process.

Figure 3 details a specific step in the evaluation process when the number of subdomains is equal to 61 (which is the number of subdomains selected by 3-wise testing). This also shows a clear advantage for GA and BO in quickly identifying low-accuracy subdomains.

Figure 4d shows that the four predictors perform similarly well. We choose Lasso as a predictor for BO because it is the best method for small training set sizes. Indeed, the beginning of the exploration, when the data is limited, is particularly important. Furthermore, it showed less variability than, for example, random forests.

4.4. Qualitative analysis of classifier failures

The main focus of our work is to efficiently detect the attributes with the most impact on classification performance. However, this subsection suggests what kind of qualitative assessment it allows. We use the BO approach and allow the exploration of 300 subdomains. Figure 5 shows the average accuracies for each attribute's value. This shows the impact of each attribute individually but does not show the impact of combinations of attributes. Figure 6 displays the impact of all the possible combinations of the attributes of weather and location.

5. Limitations

Benchmarking classifiers with generative models has limitations as observed by other work [22, 37]. There can be occasional misalignments between the prompt and the image due to bias or language limitations. For instance, in this work, we observed that the viewpoint attribute is sometimes not the one requested. We also observed generator failures for a few specific subdomains, e.g., nearly all images for "A front view of a green dog in the mountains, during the night, it is raining." are in a cartoon style, which is not the case for snowing, see Figure 7. Prompt engineering is required to allow a rigorous benchmark of the classifier. Also, generated images do not cover everything possible in the real world. Our approach tackles the computing time problem. Its main limitation is that there is no guarantee that a good selection function will identify *all* problematic subdomains for an incomplete exploration. For instance, a subdomain might be difficult for completely different reasons than the others. Thus, a selection based on learning a relation between subdomain attributes and performance might miss it.

6. Conclusion and perspectives

Text-to-Image models have great potential to be a useful tool for benchmarking image classifiers by generating images of failure cases. However, since the highest quality generators are based on diffusion models, their high inference time prevents large-scale image synthesis for advanced evaluation. This work starts from an evaluation domain described by textual attributes. To efficiently explore the critical attribute combinations that cause classifier failures, we propose to create an iterative process that alternates image generation, classifier evaluation, and attribute selection. We compare different selection functions and show that all of them outperform the method used in a previous work.

We believe that our work can be further improved by using NAS methods, taking advantage of low-fidelity evaluations. For example, in our case, the accuracy could be estimated with 20 images. The method would then use these low-fidelity evaluations to decide which combination is worth testing with high-fidelity, say 200 images. In addition, for more complex problems, one can use word embeddings such as language models instead of one-hot embeddings of finite attributes. Our work can potentially improve the benchmarking of image classifiers with text-to-image models, as it addresses a major limitation: computational time. It allows the exploration of larger domains and more precise estimates of accuracies, class probabilities, and failures.

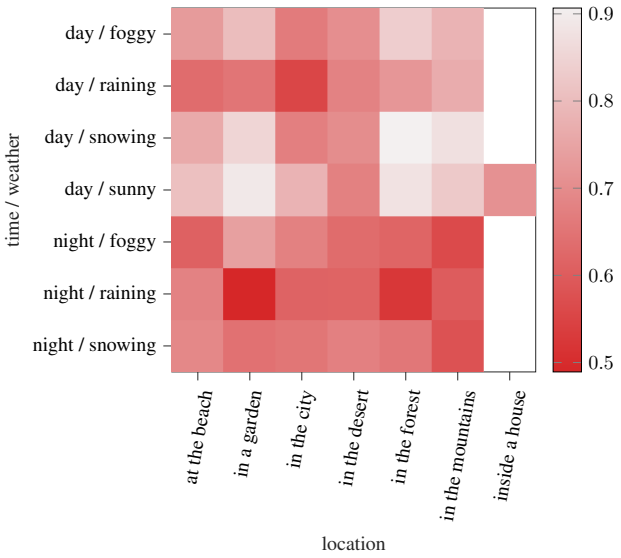


Figure 6. Heatplot displaying the average accuracies for different combinations of weather and location.

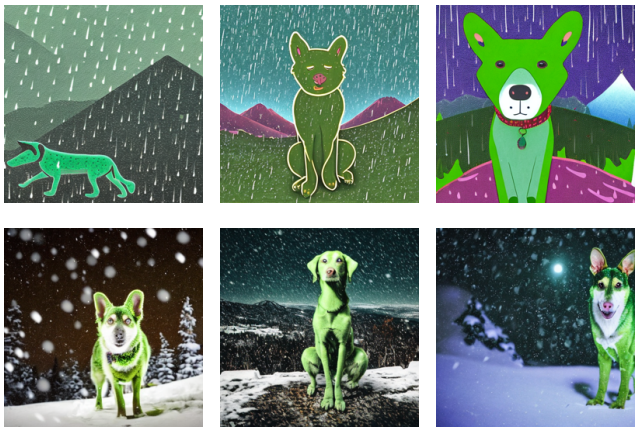


Figure 7. In the top row, images are generated with the prompt "A front view of a green dog in the mountains, during the night, it is raining.". They are mostly in a cartoon style. In the bottom row, the same prompt, but "raining" has been replaced by "snowing". The phenomenon disappears. Is this a generator failure? Careful prompt engineering, e.g., adding "a realistic image", is required to ensure alignment between the textual prompt, images, and what we expect.

Acknowledgments

This work has been supported by the French government under the "France 2030" program, as part of the SystemX Technological Research Institute.
This work was granted access to the HPC/AI resources of IDRIS under the allocation 2024-AD011013372R2 made by GENCI.

References

[1] allpairspy. https://github.com/thombashi/allpairspy. 6

[2] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016. 1

[3] J. Blank and K. Deb. pymoo: Multi-objective optimization in python. *IEEE Access*, 8:89497–89509, 2020. 6

[4] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001. 7

[5] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013. 7

[6] Krzysztof Czarnecki. Operational design domain for automated driving systems. *Taxonomy of Basic Terms ", Waterloo Intelligent Systems Engineering (WISE) Lab, University of Waterloo, Canada*, 2018. 1

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 3

[8] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 2

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 5

[10] Harris Drucker, Christopher J. C. Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. Support vector regression machines. In *Advances in Neural Information Processing Systems*. MIT Press, 1996. 7

[11] Irena Gao, Gabriel Ilharco, Scott Lundberg, and Marco Tulio Ribeiro. Adaptive testing of computer vision models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4003–4014, 2023. 2

[12] Roman Garnett. *Bayesian Optimization*. Cambridge University Press, 2023. 3

[13] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020. 1

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2

[17] John H Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992. 5

[18] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization: 5th International Conference, LION 5, Rome, Italy, January 17-21, 2011. Selected Papers 5*, pages 507–523. Springer, 2011. 3

[19] Donald R Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization*, 21:345–383, 2001. 3

[20] Kirthevasan Kandasamy, Willie Neiswanger, Jeff Schneider, Barnabas Poczos, and Eric P Xing. Neural architecture search with bayesian optimisation and optimal transport. *Advances in neural information processing systems*, 31, 2018. 2

[21] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[22] Jan Hendrik Metzen, Robin Hutmacher, N Grace Hua, Valentyn Boreiko, and Dan Zhang. Identification of systematic errors of image classifiers on rare subgroups. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5064–5073, 2023. 1, 2, 6, 8

[23] Jonas Mockus, Vytautas Tiesis, and Antanas Zilinskas. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2(117-129):2, 1978. 5

[24] Changhai Nie and Hareton Leung. A survey of combinatorial testing. *ACM Computing Surveys (CSUR)*, 43(2):1–29, 2011. 1, 6

[25] 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 *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 4

[26] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1 (2):3, 2022. 1, 2

[27] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 1

[28] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 2, 5

[29] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022. 1, 2

[30] Axel Sauer, Tero Karras, Samuli Laine, Andreas Geiger, and Timo Aila. Stylegan-t: Unlocking the power of gans for fast large-scale text-to-image synthesis. In *International conference on machine learning*, pages 30105–30118. PMLR, 2023. 2

[31] Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In *International conference on machine learning*, pages 2171–2180. PMLR, 2015. 3

[32] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015. 2

[33] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 2

[34] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996. 7

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. 3

[36] Joshua Vendrow, Saachi Jain, Logan Engstrom, and Aleksander Madry. Dataset interfaces: Diagnosing model failures using controllable counterfactual generation. *arXiv preprint arXiv:2302.07865*, 2023. 1, 2

[37] Olivia Wiles, Isabela Albuquerque, and Sven Gowal. Discovering bugs in vision models using off-the-shelf image generation and captioning. In *NeurIPS ML Safety Workshop*, 2022. 1, 2, 8

[38] Martin Zaefferer. Surrogate models for discrete optimization problems. 2018. 3