Diffusion Visual Counterfactual Explanations

Maximilian Augustin[*] **Valentyn Boreiko**[*] **Francesco Croce** **Matthias Hein**
University of Tübingen

Abstract

Visual Counterfactual Explanations (VCEs) are an important tool to understand the decisions of an image classifier. They are "small" but "realistic" semantic changes of the image changing the classifier decision. Current approaches for the generation of VCEs are restricted to adversarially robust models and often contain non-realistic artefacts, or are limited to image classification problems with few classes. In this paper, we overcome this by generating Diffusion Visual Counterfactual Explanations (DVCEs) for arbitrary ImageNet classifiers via a diffusion process. Two modifications to the diffusion process are key for our DVCEs: first, an adaptive parameterization, whose hyperparameters generalize across images and models, together with distance regularization and late start of the diffusion process, allow us to generate images with minimal semantic changes to the original ones but different classification. Second, our cone regularization via an adversarially robust model ensures that the diffusion process does not converge to trivial non-semantic changes, but instead produces realistic images of the target class which achieve high confidence by the classifier. Code is available under `https://github.com/valentyn1boreiko/DVCEs`.

1 Introduction

It can be argued that one of the main problems hindering the widespread use of machine learning, and image classification in particular, is the missing possibility to explain the decisions of black-box models such as neural networks. This is not only a problem for decisions affecting humans where the current draft for AI regulation in Europe [10] requires "transparency", but it is a pressing problem in all applications of machine learning. The reason, to some extent, is that humans would like to understand but also control if the learning algorithm has captured the "concepts" of the underlying classes or if it just predicts well using spurious features, artefacts in the data set, or other sources of error. In this paper, we focus on model-agnostic explanations which can, in principle, be applied to any image classifier and do not rely on the specific structure of the classifier such as decision trees or linear classifiers. In this area, in particular for image classification, sensitivity based explanations [4], explanations based on feature attributions [3], saliency maps [47, 46, 17, 56, 51], Shapley additive explanations [31], and local fits of interpretable models [39] have been proposed. Moreover, [55] proposed counterfactual explanations (CEs), which are instance-specific explanations. They can be applied to any classifier and it has been argued that they are close to the human justification of decisions [34] using counterfactual reasoning: "I would recognize it as zebra (instead of horse) if it had black and white stripes." For a given classifier almost all methods for the generation of CEs [55, 15, 35, 6, 38, 54, 44] try to solve the following problem: "Given a target class c what is the minimal change δ of input x, such that $x + \delta$ is classified as class c with high probability and is a realistic instance of my data generating distribution?" From the perspective of "debugging" existing machine learning models, CEs are interesting as they construct a concrete input $x + \delta$ with a different classification that allows the developer to test if the model has learned the correct features.

[*]Equal contribution.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

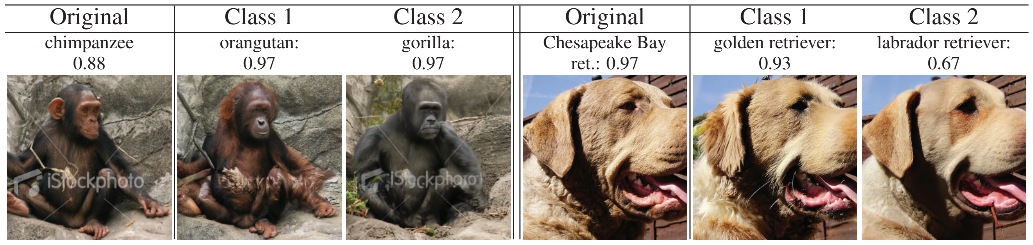


Figure 1: Our method DVCE creates visual counterfactual explanations (VCEs) that explain any given classifier (here: non-robust ConvNeXt) by changing an image of one class into a similar target class by changing class relevant features while preserving the overall image structure.

The main reason why visual counterfactual explanations (VCEs), that is CEs for image classification, are not widely used is that the tasks of generating CEs and adversarial examples [52] are very related. Even imperceivable changes of the image can already change the prediction of the classifier, however, the resulting noise patterns do not show the user if the classifier has picked up the right class-specific features. One possible solution is using adversarially robust models [43, 1, 7], which have been shown to produce semantically meaningful VCEs by directly maximizing the probability of the target class in image space. These approaches have the downside that they only can generate VCEs for robust models which is a significant restriction as these models are not competitive in terms of prediction accuracy. The second approach is to restrict the generation of VCEs using a generative model or constraining the set of potential image manipulations [19, 20, 42, 8, 18, 45, 27]. However, these approaches are either restricted to datasets with a small number of classes, cannot provide explanations for arbitrary classifiers, or generate VCEs that look realistic but have so little in common with the original image that not much insight can be gained.

Recently, [26] trained a StyleGAN2 [24] model to discover and manipulate class attributes. While their approach yields impressive results on smaller-scale datasets with few similar classes (for example, different bird species), the authors did not demonstrate that the method scales to complex tasks such as ImageNet with hundreds of classes where different classes require different sets of attributes. Another disadvantage is that the StyleGAN model needs to be retrained for every classifier, making it prohibitively expensive to explain multiple large models. Moreover, [25] have proposed a loss to do VCEs in the latent space of a GAN/VAE. They show promising results for MNIST/FMNIST, but no code is available for ImageNet. Furthermore, to explain a classifier, the conditional information during the generation of explanations should ideally come only from the classifier itself, but [25] rely on a conditional GAN which might introduce a bias.

In this paper, we overcome the aforementioned challenges and generate our Diffusion Visual Counterfactual Explanations (DVCEs) for arbitrary ImageNet classifiers (see Fig 1). We use the progress in the generation of realistic images using diffusion processes [48, 23, 49, 50], which recently were able to outperform GANs [14]. Similar to [37, 2], we use a classifier and a distance-type regularization to guide the generation of the images. Two modifications to the diffusion process are key elements for our DVCEs: i) a combination of distance regularization and starting point of the diffusion process together with an adaptive reparameterization lets us generate VCEs visually close to the original image in a controlled way so that hyperparameters can be fixed across images and even across models, ii) our cone regularization of the gradient of the classifier via an adversarially robust model ensures that the diffusion process does not converge to trivial non-semantic changes but instead produces realistic images of the target class which achieve high confidence by the classifier. Our approach can be employed for any dataset where a generative diffusion model and an adversarially robust model are available. In a qualitative comparison, user study, and a quantitative evaluation (Sec. 4.1), we show that our DVCEs achieve higher realism (according to FID) and have more meaningful features (according to the user study) than both the recent methods of [7] and [2].

2 Diffusion models

Diffusion models [14, 48, 49, 23, 36] are generative models that consist of two steps: a *forward diffusion* process (and thus a Markov process) that transforms the data distribution to a prior distribution (which is usually assumed to be a standard normal distribution) and the *reverse diffusion* process that

transforms the prior distribution back to the data distribution. The existence of the (continuous-time) reverse diffusion process was closer investigated in [50].
In the discrete-time setting, a Markov chain $\{x_1, ..., x_T\}$ for any data point x_0 in the forward direction is defined via a Markov process that adds noise to the data point at each timestep:

$$q(x_t|x_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t I), \tag{1}$$

where $\{\beta_1, ..., \beta_T\}$ is some variance schedule, chosen such that $q(x_T) \sim \mathcal{N}(0, I)$. Note that given x_0, it is possible to sample from $q(x_t|x_0)$ in closed form instead of applying noise t times using:

$$q(x_t|x_0) = \mathcal{N}(\sqrt{\overline{\alpha}_t}x_0, 1 - \overline{\alpha}_t I), \tag{2}$$

$$x_t = \sqrt{\overline{\alpha}_t}x_0 + \epsilon\sqrt{1-\overline{\alpha}_t}, \quad \epsilon \sim \mathcal{N}(0, I), \tag{3}$$

where $\overline{\alpha}_t := \prod_{k=1}^{t}(1-\beta_k)$. In [48], the authors have shown that the reverse transitions $q(x_{t-1}|x_t)$ approach diagonal Gaussian distributions as $T \to \infty$ and thus one can use a DNN with parameters θ to approximate $q(x_{t-1}|x_t)$ as $p_\theta(x_{t-1}|x_t)$ by predicting the mean and the diagonal covariance matrix $\mu_\theta(x_t, t)$ and $\Sigma_\theta(x_t, t)$:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(\mu_\theta(x_t, t), \Sigma_\theta(x_t, t)). \tag{4}$$

To sample from $p_\theta(x_0)$, one samples x_T from $\mathcal{N}(0, I)$ and then follows the reverse process by repeatedly sampling from the transition probabilities $p_\theta(x_{t-1}|x_t)$.

Instead of predicting $\mu_\theta(x_t, t)$ directly, it has been shown in [23] that the best performance can be achieved when a neural network $\epsilon_\theta(x_t, t)$ predicts the source noise ϵ in (3). Then, the loss used for training resembles that of a denoising model:

$$L_{\text{simple}}(\theta, T, q(x_0)) := \mathbb{E}_{t\sim[1,T], x_0\sim q(x_0), \epsilon\sim\mathcal{N}(0,I)} \left\|\epsilon - \epsilon_\theta(x_t, t)\right\|^2. \tag{5}$$

The mean of the reverse step μ_θ in (4) can be derived [32] using Bayes theorem as:

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{1-\beta_t}}\Big(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)\Big). \tag{6}$$

This is, however, only one of the possible equivalent parameterization of the learning objective as shown in [32]. Because the objective L_{simple} does not give a learning signal for $\Sigma_\theta(x_t, t)$, in practice one combines L_{simple} with another loss based on a variational lower bound of the data likelihood [48], which unlike L_{simple} allows us to learn the diagonal covariance matrix $\Sigma_\theta(x_t, t)$. Concretely, the network $\epsilon_\theta(x_t, t)$ outputs additionally a vector that is used to parametrize $\Sigma_\theta(x_t, t)$.

2.1 Class conditional sampling

There exist diffusion models trained with and without knowledge about classes in the dataset [14, 22]. For our experiments, we only use the class-unconditional diffusion model, such that all class-conditional features are introduced by the classifier that we want to explain. For the noise-aware classifier $p_\phi(y|\cdot, \cdot) : \mathbb{R}^d \times \{1, ..., T\} \to [0, 1]$, with parameters ϕ, that is trained on noisy images corresponding to the various timesteps [14], the reverse process transitions are then of the form:

$$p_{\theta,\phi}(x_{t-1}|x_t, y) = Z\, p_\theta(x_{t-1}|x_t)\, p_\phi(y|x_t, t) \tag{7}$$

for a normalization constant Z. As we would like to explain any classifier and not only noise-aware ones, we follow the approach of [2], where a classifier $p_\phi(y|\cdot) : \mathbb{R}^d \to [0, 1]$ is given as input the denoised sample $\hat{x}_0 = f_{\text{dn}}(x_t)$ of x_t, using the mapping:

$$f_{\text{dn}} : \mathbb{R}^d \times \{1, ..., T\} \to \mathbb{R}^d, \quad (x_t, t) \mapsto \frac{x_t}{\sqrt{\overline{\alpha}_t}} - \frac{\sqrt{1-\overline{\alpha}_t}\epsilon_\theta(x_t, t)}{\sqrt{\overline{\alpha}_t}}. \tag{8}$$

This mapping, derived from (3), estimates the noise-free image x_0 using the noise approximated by the model $\epsilon_\theta(x_t, t)$ for a given timestep t. With this, we can define a timestep-aware posterior $p_\phi(y|x_t, t)$ for any classifier $p_\phi(y|\cdot)$ as $p_\phi(y|x_t, t) := p_\phi(y|f_{\text{dn}}(x_t, t))$.

The issue is that efficient sampling from the original diffusion model is possible only because the reverse process is made of normal distributions. As we need to sample from $p_{\theta,\phi}(x_{t-1}|x_t, y)$ hundreds of times to obtain a single sample from the data distribution, it is not possible to use MCMC-samplers with high complexity to sample from each of the individual transitions. In [14], they proposed to solve it by approximating $p_{\theta,\phi}(x_{t-1}|x_t, y)$ with slightly shifted versions of $p_\theta(x_{t-1}|x_t)$ to make closed-form sampling possible. Such transition kernels are given by:

$$p_{\theta,\phi}(x_{t-1}|x_t, y) = \mathcal{N}(\mu_t, \Sigma_\theta(x_t, t)), \tag{9}$$

$$\mu_t = \mu_\theta(x_t, t) + \Sigma_\theta(x_t, t)\nabla_{x_t} \log p_\phi(y|x_t, t), \tag{10}$$

which we further adapt for the goal of generating VCEs and use in our experiments.

3 Diffusion Visual Counterfactual Explanations

A VCE x for a chosen target class y, a given classifier $p_\phi(y|\cdot)$, and an input $\hat{x}$ should satisfy the following criteria: i) **validity:** the VCE x should be classified by $p_\phi(y|\cdot)$ as the desired target class y with high predicted probability, ii) **realism:** the VCE should be as close as possible to a natural image, iii) **minimality/closeness:** the difference between the VCE x and the original image $\hat{x}$ should be the minimal semantic modification necessary to change the class, in particular, the generated image x should be close to $\hat{x}$ while being valid and realistic, e.g. by changing the object in the image and leaving the background unchanged. Note that targeted adversarial examples are valid but do not show meaningful semantic changes in the target class for a non-robust model and are not realistic. The $l_{1.5}$-SVCEs of [7] change the image in order to maximize the predicted probability of the classifier into the target class inside an $l_{1.5}$-ball around the image which is a targeted adversarial example. Thus this only works for robust classifiers and they use an ImageNet classifier that was trained to be multiple-norm robust (MNR), which we denote in this paper as MNR-RN50 (see Sec. 4.2). The realism of the $l_{1.5}$-SVCEs comes purely from the generative properties of robust classifiers [43], which can lead to artefacts. In contrast, our Diffusion Visual Counterfactual Explanations (DVCEs) work for any classifier and our DVCEs are more realistic due the better generative properties of diffusion models. An approach similar to our DVCE framework is Blended Diffusion (BD) [2] which manipulates the image inside a masked region. One can adapt BD for the generation of VCEs by using as mask the whole image. DVCE and BD share the same diffusion model, but BD cannot be applied to arbitrary classifiers and requires image-specific hyperparameter tuning, see Fig. 2.

3.1 Adaptive Parameterization

For generating the DVCE of the original image $\hat{x}$, we do not want to sample just any image x from $p(x|y)$ (high realism and validity), but also to make sure that $d(x, \hat{x})$ is small. For this, we have to condition our diffusion process on $\hat{x}$. Using the denoising function f_{dn} and analogously to the derivation of (9) and its applications in [14, 27], the mean of the transition kernel becomes:

$$\mu_\theta(x_t, t) + \Sigma_\theta(x_t, t)\nabla_{x_t}\Big[C_c \log p_\phi\big(y|f_{\text{dn}}(x_t, t)\big) - C_d\, d\big(\hat{x}, f_{\text{dn}}(x_t, t)\big)\Big], \tag{11}$$

where C_c is the coefficient of the classifier, and C_d the one of the distance guiding loss. Intuitively, we take a step in the direction that increases the classifier score while staying close to $\hat{x}$. Note that the last term can be interpreted as the log gradient of a distribution with density $\exp(-C_d\, d_t(\hat{x}, \cdot))$, where $d_t(\hat{x}, \cdot) := d(\hat{x}, f_{\text{dn}}(\cdot, t))$. Thus we introduce a timestep-aware prior distribution that enforces our output to be similar to $\hat{x}$. In our work, we use the l_1-distance as it produces sparse changes.

Several works have tried to minimize some distance during the diffusion process, implicitly in [9] or explicitly, but for the background, in BD. However, there is no principled way to choose the coefficient for such a regularization term. It turns out that it is impossible to find a parameter setting for BD that works across images and classifiers. Thus, we propose a parameterization

$$g_{\text{update}} = C_c \frac{\nabla_{x_t} \log p_\phi(y|f_{\text{dn}}(x_t, t)))}{\|\nabla_{x_t} \log p_\phi(y|f_{\text{dn}}(x_t, t)))\|_2} - C_d \frac{\nabla_{x_t} d(\hat{x}, f_{\text{dn}}(x_t, t))}{\|\nabla_{x_t} d(\hat{x}, f_{\text{dn}}(x_t, t))\|_2}, \tag{12}$$

which adapts to the predicted mean of the diffusion model that we use to change μ_t in (10) to

$$\mu_t = \mu_\theta(x_t, t) + \Sigma_\theta(x_t, t)\, \|\mu_\theta(x_t, t)\|_2\, g_{\text{update}}. \tag{13}$$

This adaptive parameterization allows for fine-grained control of the influence of the classifier and distance regularization so that now the hyperparameters C_c and C_d have the same influence across

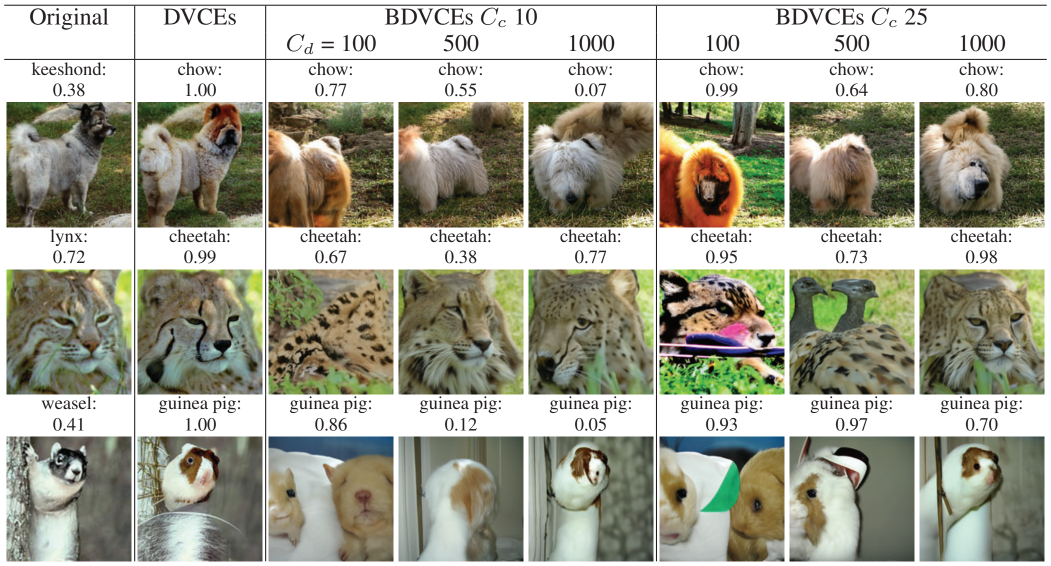


Figure 2: DVCEs for the robust MNR-RN50 model [7] in the second column (ours) and using Blended Diffusion (BDVCEs) [2], such that regularization is applied to the whole image. Due to our adaptive parameterization, the same parameters ($C_c = 0.1, C_d = 0.15$) work for our DVCEs across different images and classifiers (see Fig. 5). For BDVCEs, it is difficult to choose a set of hyperparameters that, even for a single classifier, work for multiple images. For BDVCEs, we report the parameter C_d of the LPIPS weight and the l_2-weight is 10 times as large (ratio used in [2]).

images and even classifiers. It facilitates the generation of DVCEs as otherwise hyperparameter finetuning would be necessary for each image as in BD, see Fig. 2 for a comparison. However, even with our adaptive parameterization, it is still not easy to produce semantically meaningful changes close to $\hat{x}$ as can be seen in App. B.2. Thus, as in [2], we vary the starting point of the diffusion process and observe in App. B.2 that starting from step $\frac{T}{2}$ of the forward diffusion process, together with the adaptive parameterization and using as the distance the l_1-distance, provides us with sparse but semantically meaningful changes. In our experiments, we set $T = 200$.

3.2 Cone Projection for Classifier Guidance

A key objective for our DVCEs is that they can be applied to any image classifier, adversarially robust or not. Diffusion with the new mean as in (13) does not work with a non-robust classifier and leads to very small modifications of the image, which are similar to adversarial examples without meaningful changes. The reason is that the gradients of non-robust classifiers are noisier and less semantically meaningful than those of robust classifiers. We illustrate this for a non-robust Swin-TF [28] in Fig. 3 where hardly any changes are generated. This even holds if we first denoise the sample using f_{dn} from (8) and average gradients of augmented images as in [2] (see Appendix B.6 for details).

As a solution, we suggest projecting the gradient of an adversarially robust classifier with parameters ψ, $\nabla_{x_t} \log p_{\text{robust},\psi}(y|f_{\text{dn}}(x_t,t))$, onto a cone centered at the gradient of the classifier, $\nabla_{x_t} \log p_\phi(y|f_{\text{dn}}(x_t,t))$. More precisely, we define

$$g_{\text{proj}} = P_{\text{cone}(\alpha, \nabla_{x_t} \log p_\phi(y|f_{\text{dn}}(x_t,t)))}\Big[\nabla_{x_t} \log p_{\text{robust},\psi}\big(y|f_{\text{dn}}(x_t,t)\big)\Big],$$

where $\text{cone}(\alpha, v) := \{w \in \mathbb{R}^d : \angle(v,w) \le \alpha\}$ is the cone of angle α around vector v, and the projection $P_{\text{cone}(\alpha,v)}$ onto $\text{cone}(\alpha,v)$ is given as:

$$P_{\text{cone}(\alpha,v)}[w] := \begin{cases} \langle u, w\rangle\, u, & \angle(w,v) > \alpha \\ w, & \text{else,} \end{cases}$$

where $P_{v^\perp}(w) := w - \frac{\langle w,v\rangle}{\langle v,v\rangle} v$ and $u = \sin(\alpha)\frac{P_{v^\perp}(w)}{\left\|P_{v^\perp}(w)\right\|_2} + \cos(\alpha)\frac{v}{\|v\|_2}$ (note $\|u\|_2 = 1$). The motivation for this projection is to reduce the noise in the gradient of the non-robust classifier. Note

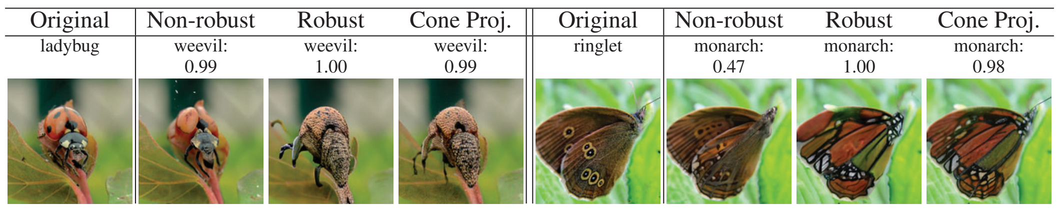


Figure 3: DVCEs for the non-robust Swin-TF[28], robust MNR-RN50 model, and cone-projected DVCEs for Swin-TF[28]. Similar to adversarial examples, guidance by the non-robust Swin-TF does not yield semantically meaningful changes. In contrast, the DVCEs of the robust MNR-RN50 and the DVCE with cone projection for the Swin-TF[28] yield valid and realistic VCEs.

that the projection of the gradient of the robust classifier onto the cone generated by the non-robust classifier with angle $\alpha < 90°$ is always an ascent direction for $\log p_\phi(y|f_{\text{dn}}(x_t,t))$, which we would like to maximize (note that g_{proj} is not necessarily an ascent direction for $\log p_{\text{robust},\psi}(y|f_{\text{dn}}(x_t,t))$). Thus, the cone projection is a form of regularization of $\nabla_{x_t} \log p_\phi(y|f_{\text{dn}}(x_t,t))$, which guides the diffusion process to semantically meaningful changes of the image. The averaging over augmentations of x_t has an additional regularizing effect, see Appendix B.6.

3.3 Final Scheme for Diffusion Visual Counterfactuals

Our solution for a non-adversarially robust classifier $p_\phi(y|\cdot)$ is to use Algorithm 1 of [14] by replacing the update step with:

$$\begin{aligned}
g_{\text{proj}} &= P_{\text{cone}(\alpha, \nabla_{x_t} \log p_\phi(y|f_{\text{dn}}(x_t,t)))}\Big[\nabla_{x_t} \log p_{\text{robust},\psi}\big(y|f_{\text{dn}}(x_t,t)\big)\Big],\\
g_{\text{update}} &= C_c \frac{g_{\text{proj}}}{\|g_{\text{proj}}\|_2} - C_d \frac{\nabla_{x_t} d(\hat{x}, f_{\text{dn}}(x_t,t))}{\|\nabla_{x_t} d(\hat{x}, f_{\text{dn}}(x_t,t))\|_2},\\
\mu_t &= \mu_\theta(x_t,t) + \Sigma_\theta(x_t,t)\,\|\mu_\theta(x_t,t)\|_2\, g_{\text{update}},\\
p(x_{t-1}|x_t,\hat{x},y) &= \mathcal{N}(\mu_t, \Sigma_\theta(x_t,t)).
\end{aligned}$$

For an adversarially robust classifier the cone projection is omitted and one uses $g_{\text{proj}} = \nabla_{x_t} \log p_{\text{robust},\psi}\big(y|f_{\text{dn}}(x_t,t)\big)$. In all our experiments we use $C_c = 0.1$, and $C_d = 0.15$ unless we show ablations for one of the parameters. The angle α for the cone projection is fixed to 30°. In strong contrast to BD, these parameters generalize across images and classifiers.

4 Experiments

In this section, we evaluate the quality of the DVCE. We compare DVCE to existing works in Sec. 4.1. In Sec. 4.2, we compare DVCEs for various state-of-the-art ImageNet models and show how DVCEs can be used to interpret differences between classifiers. For our DVCEs, we use for all experiments the fixed hyperparameters from Sec. 3.3. The diffusion model used for DVCE is from [14] and was trained class-unconditionally on 256x256 ImageNet images using a modified UNet[40]. A user study, discovery of spurious features using VCEs, further experiments, and ablations are in the appendix.

4.1 Comparison of Methods for VCE Generation

We compare DVCEs with VCEs produced by BD (BDVCEs) and $l_{1.5}$-SVCEs. As the latter only works for adversarially robust classifiers, we use the multiple-norm robust ResNet50 from [7], MNR-RN50, as the classifier to create VCEs for all three methods. As this model is robust on its own, we do not use the cone projection for its DVCEs.

First, in Fig. 4 we present a qualitative evaluation, where we transform one image into two different classes that are close to the true one in the WordNet hierarchy[33]. The radius of the $l_{1.5}$-ball of [7] is chosen as the smallest $r \in \{50, 75, 100, 150\}$ such that the confidence in the target class is larger than 0.9 per image. For BD, we select the image with the smallest classifier and regularization weight that reaches confidence larger than 0.9 from the set of parameters discussed in Sec. 3.3. If 0.9 is not reached by any setting for one of the two baselines, we show the image that achieves the highest

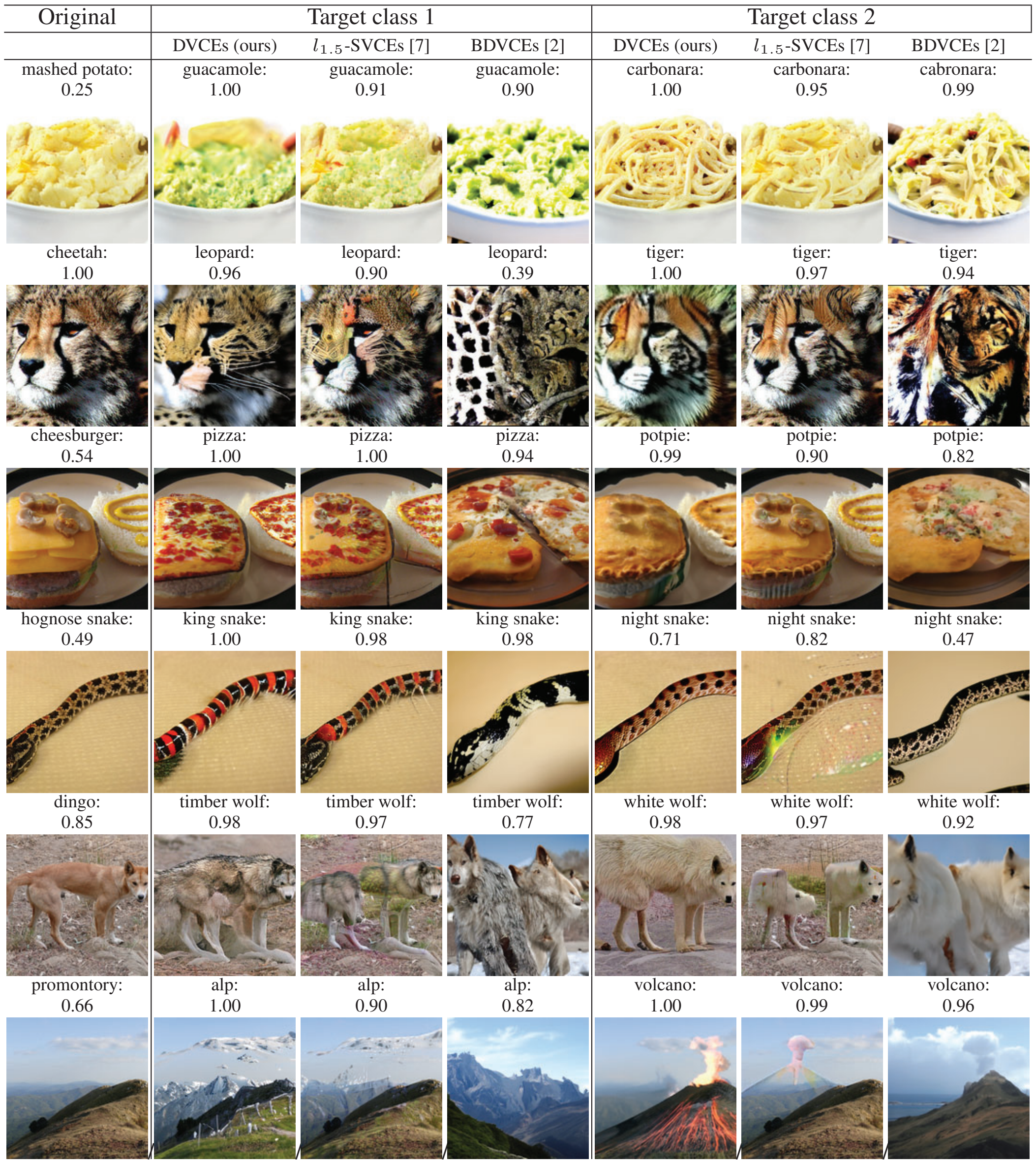


Figure 4: Comparison of different VCE-methods for the robust MNR-RN50 (taken from [7]). We show for each original image (outer left column) and for each target class our DVCEs (left column), the $l_{1.5}$-SVCEs of [7] (middle), and the adaptation of BDVCEs [2] (right). Only DVCEs satisfy all desired properties of VCEs. BD fails for leopard and tiger and often produces images far away from the original one (pizza, potpie, timber wolf, white wolf, and alp). $l_{1.5}$-SVCEs show artefacts (white wolf, volcano, night snake) and have often lower image quality.

confidence. As Fig. 4 shows, DVCE is the only method that satisfies all desired properties of VCEs. For example, for “mashed potato”, DVCE preserves the bowl and only changes the content into either guacamole or carbonara. Our qualitative comparison shows that the same hyperparameter setting of DVCE can handle different classes and transfer between similar classes, such as different snakes or wolf types, as well as different object sizes, e.g. cheetah and snake.

In contrast, both $l_{1.5}$-SVCEs and BDVCEs require different hyperparameters for different images to achieve high confidence in the target class. Even with the six parameter configurations, BD is not able to always produce images with high confidence in the target class. More problematic for VCEs is that the resulting images can have high confidence but are neither realistic (cheetah $\rightarrow$ tiger) nor resemble the original image at all (dingo $\rightarrow$ timber wolf/white wolf). Even if the method works with the given parameters, for example, mashed potato $\rightarrow$ guacamole or carbonara, the overall image quality cannot match that of DVCE as often the images contain overly bright colors. In the case of the volcano VCE, DVCE shows class features like lava whereas BDVCE can not clearly be labeled as a volcano. For the $l_{1.5}$-SVCEs, one often needs large radii to achieve the desired confidence of 0.9, which often results in images that do not look realistic.

As noted in [7], a quantitative analysis of VCEs using FID scores is difficult as methods not changing the original image have low FID score. Thus, we have developed a cross-over evaluation scheme, where one partitions the classes into two sets and only analyzes cross-over VCEs (more details are in App. E). We show the results in Tab. 1. In terms of closeness, DVCEs are worse than $l_{1.5}$-SVCEs, which can be expected as they optimize inside an $l_{1.5}$-ball. However, DVCE are the most realistic ones (FID score) and have similar validity as $l_{1.5}$-SVCEs. BDVCEs are the worst in all categories.

Moreover, we have conducted a user study (20 users), in which participants decided if the changes of the VCE are **meaningful** or **subtle** and if the generated image is **realistic**, see App. C for details. The percentage of total images having the three different properties is (order: DVCEs, $l_{1.5}$-SVCEs, BD): **meaningful** - **62.0%**, 48.4%, 38.7%; **realism** - 34.7%, 24.6%, **52.2%**; **subtle** - 45.0%, **50.6%**, 31.0%. This confirms that DVCEs generate more meaningful features in the target classes. While the result regarding realism seems to contradict the quantitative evaluation, this is due to fact that realism means that the user considered the image realistic irrespectively if it shows the target class or not.

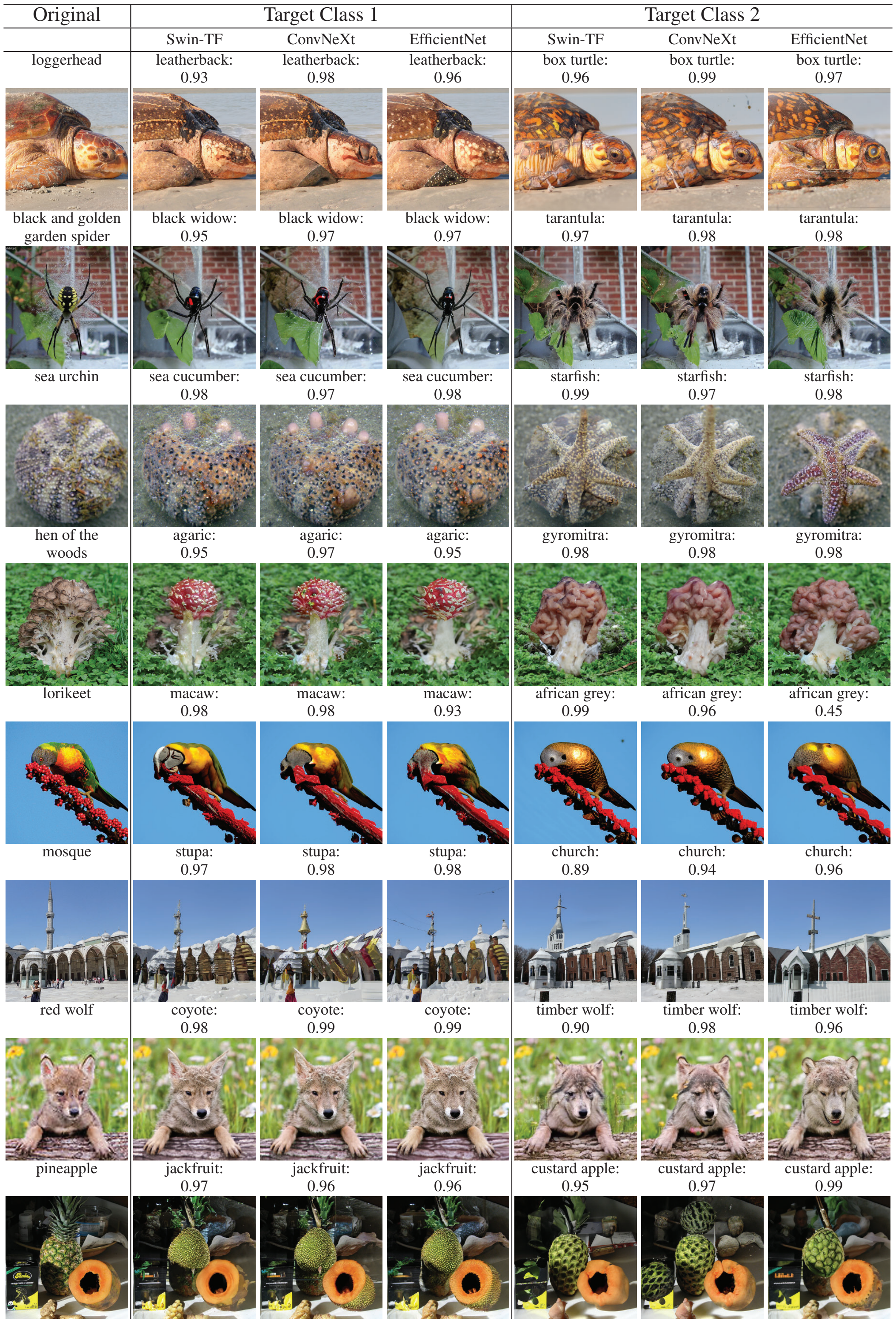
	Closeness				**Validity**	**Realism**
Metric	$l_1 \downarrow$	$l_{1.5} \downarrow$	$l_2 \downarrow$	LPIPS-Alex $\downarrow$	Mean Conf. $\uparrow$	Avg. FID $\downarrow$
DVCEs (ours)	12799	293	48	0.35	0.932	**17.6**
$l_{1.5}$-SVCEs[7]	**5139**	**139**	**26**	**0.20**	**0.945**	25.6
Blended Diffusion[2]	35678	722	108	0.58	0.825	27.9

Table 1: Quantitative evaluation of VCEs. DVCEs outperform BDVCEs [2] in all metrics. Moreover, they achieve comparable to $l_{1.5}$-SVCEs Mean Conf. (validity), while outperforming them significantly in Avg. FID. (realism) but do larger changes to the image than SVCE (closeness).

4.2 Model comparison

Non-robust models. In Fig. 5 we show that DVCEs can be generated for various state-of-the-art ImageNet [41] models. We use a Swin-TF-L[28], a ConvNeXt-L [29] and a Noisy-Student EfficientNet-B7 [57, 53]. Both the Swin-TF and the ConvNeXt are pretrained on ImageNet21k [13] whereas the EfficientNet uses noisy-student self-training on a large unlabeled pool. As all models do not yield perceptually aligned gradients, we use the cone projection with 30° angle described in Sec. 3.2 with the robust model from the previous section. All other parameters are identical to the previous experiment. We highlight that DVCE is the first method capable of explaining (in the sense of VCEs) arbitrary classifiers on a task as challenging as ImageNet. Overall, DVCEs satisfy all desired properties of VCEs. It also allows us to inspect the most important features of each model and class. For the stupa and church classes, for example, it seems like all models use different roof and tower structures as the most prominent feature as they spent most of their budget on changing those.

Robust models. Here, we evaluate the DVCEs of different robust models trained to be multiple-norm adversarially robust. They are generated by multiple norm-finetuning [11] an initially l_p-robust model to become robust with respect to l_1-, l_2- and l_∞-threat models. Specifically an l_2-robust ResNet50 [16] results in MNR-RN50 used in the previous sections, an l_∞-robust XCiT-transformer model [12] - in MNR-XCiT in the following and an l_∞-robust DeiT-transformer [5] - in MNR-DeiT. Their multiple-norm robust accuracies can be found in [11]. In [7], different l_p-norm adversarially robust models were compared for the generation of VCEs and they showed that for their $l_{1.5}$-SVCEs the multiple norm robust model was better (both in terms of FID and qualitatively) than individual l_p-norm robust classifiers. This is the reason why we also use a multiple-norm robust model for the cone-projection of a non-robust classifier. In Fig. 6, we show DVCEs of the three different classes for two examples images and two target classes each. All of the multiple-norm robust models have similarly good DVCEs showing classifier-specific variations in the semantic changes. This proves again the generative properties of adversarially robust models, in particular of robust transformers. More examples and further experiments are in App. B.4.



Original	Target Class 1			Target Class 2		
	Swin-TF	ConvNeXt	EfficientNet	Swin-TF	ConvNeXt	EfficientNet
loggerhead	leatherback: 0.93	leatherback: 0.98	leatherback: 0.96	box turtle: 0.96	box turtle: 0.99	box turtle: 0.97
black and golden garden spider	black widow: 0.95	black widow: 0.97	black widow: 0.97	tarantula: 0.97	tarantula: 0.98	tarantula: 0.98
sea urchin	sea cucumber: 0.98	sea cucumber: 0.97	sea cucumber: 0.98	starfish: 0.99	starfish: 0.97	starfish: 0.98
hen of the woods	agaric: 0.95	agaric: 0.97	agaric: 0.95	gyromitra: 0.98	gyromitra: 0.98	gyromitra: 0.98
lorikeet	macaw: 0.98	macaw: 0.98	macaw: 0.93	african grey: 0.99	african grey: 0.96	african grey: 0.45
mosque	stupa: 0.97	stupa: 0.98	stupa: 0.98	church: 0.89	church: 0.94	church: 0.96
red wolf	coyote: 0.98	coyote: 0.99	coyote: 0.99	timber wolf: 0.90	timber wolf: 0.98	timber wolf: 0.96
pineapple	jackfruit: 0.97	jackfruit: 0.96	jackfruit: 0.96	custard apple: 0.95	custard apple: 0.97	custard apple: 0.99

Figure 5: DVCEs for three non-robust classifiers: Swin-TF[28], ConvNeXt[29] and EfficientNet[57]. For each original image (outer left column) we show DVCEs into two target classes. Please zoom into the images to see more fine-grained details and differences.

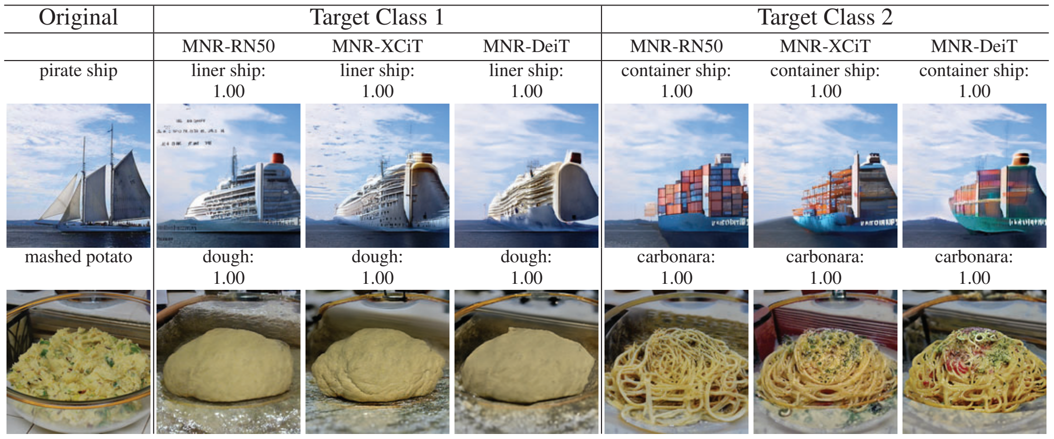


Figure 6: We compare DVCEs for three different robust models (no cone projection) which are all fine-tuned to be multiple-norm adversarially robust [11], that is against l_1, l_2 and l_∞-perturbations. More examples can be found in Fig. 10.

5 Limitations, Future Work and Societal impact

In comparison to GANs, diffusion-based approaches can be expensive to evaluate as they require multiple iterations of the reverse process to create one sample. This can be an issue when creating a large amount of VCEs and makes deployment in a time-sensitive setting challenging.

We also rely on a robust model during the creation of VCEs for the cone projection. While we show that standard classifiers do not yield the desired gradients, training robust models can be challenging. An interesting direction for future research is thus to replace the cone projection with another "denoising" procedure for the gradient of a non-robust model.

From a theoretical standpoint, using conditional sampling with reverse transitions of the form (7) is justified, in practice, however, we have to approximate the reverse transitions with shifted normal distributions of the form (9). This approximation relies on the conditioning function having low curvature, and it can be hard to verify this once we start adding a classifier, distance and other possible terms and it is unclear how this influences the outcome of the diffusion process.

The quantitative evaluation of VCEs is difficult, as the standard FID metric compares the distribution of features of a classifier over a test and a generated dataset. However, for VCEs it is not only important to generate realistic images, but also to achieve high confidence and to create meaningful changes. Moreover, metrics such as IM1, IM2 [30] for VCEs rely on a well-trained (V)AE for every class, which is difficult to achieve for a dataset with 1000 classes and high-resolution images. Future research should therefore try to develop metrics for the quantitative evaluation of VCEs and we think that the evaluation in Tab. 1 is the first step in that direction.

DVCEs and VCEs in general help to discover biases of the classifiers and thus have a positive societal impact, however one can abuse them for unintended purposes as any conditional generative model.

6 Conclusion

We have proposed DVCEs, a novel way to create VCEs for any state-of-the-art ImageNet classifier using diffusion models and our cone projection. DVCEs can handle vastly different image configurations, object sizes, and classes and satisfy all desired properties of VCEs.

Acknowledgement

The authors acknowledge support by the DFG Excellence Cluster Machine Learning - New Perspectives for Science, EXC 2064/1, Project number 390727645 and the German Federal Ministry of Education and Research (BMBF): Tübingen AI Center, FKZ: 01IS18039A as well as the DFG grant 389792660 as part of TRR 248.

References

[1] Maximilian Augustin, Alexander Meinke, and Matthias Hein. Adversarial robustness on in- and out-distribution improves explainability. In *ECCV*, 2020.

[2] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *CVPR*, 2022.

[3] Sebastian Bach, Alexander Binder, Frederick Klauschen Gregoire Montavon, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS One*, 2015.

[4] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *JMLR*, 2010.

[5] Yutong Bai, Jieru Mei, Alan Yuille, and Cihang Xie. Are transformers more robust than cnns? In *NeurIPS*, 2021.

[6] Solon Barocas, Andrew D. Selbst, and Manish Raghavan. The hidden assumptions behind counterfactual explanations and principal reasons. In *FAT*, 2020.

[7] Valentyn Boreiko, Maximilian Augustin, Francesco Croce, Philipp Berens, and Matthias Hein. Sparse visual counterfactual explanations in image space. In *GCPR*, 2022.

[8] Chun-Hao Chang, Elliot Creager, Anna Goldenberg, and David Duvenaud. Explaining image classifiers by counterfactual generation. In *ICLR*, 2019.

[9] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models. In *ICCV*, 2021.

[10] EU Commission. Regulation for laying down harmonised rules on AI. *European Commission*, 2021.

[11] Francesco Croce and Matthias Hein. Adversarial robustness against multiple l_p-threat models at the price of one and how to quickly fine-tune robust models to another threat model. In *ICML*, 2022.

[12] Edoardo Debenedetti. Adversarially robust vision transformers. Master's thesis, Swiss Federal Institue of Technology, Lausanne (EPFL), 2022.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: a large-scale hierarchical image database. In *CVPR*, 2009. License: No license specified.

[14] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021.

[15] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In *NeurIPS*, 2018.

[16] Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 2019. License: MIT.

[17] Christian Etmann, Sebastian Lunz, Peter Maass, and Carola-Bibiane Schönlieb. On the connection between adversarial robustness and saliency map interpretability. In *ICML*, 2019.

[18] Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In *ICML*, 2019.

[19] Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generating visual explanations. In *ECCV*, 2016.

[20] Lisa Anne Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. Grounding visual explanations. In *ECCV*, 2018.

[21] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017.

[22] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS Workshop*, 2021.

[23] Pieter Abbeel Jonathan Ho, Ajay Jain. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[24] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *CVPR*, 2020.

[25] Saeed Khorram and Li Fuxin. Cycle-consistent counterfactuals by latent transformations. In *CVPR*, 2022.

[26] Oran Lang, Yossi Gandelsman, Michal Yarom, Yoav Wald, Gal Elidan, Avinatan Hassidim, William T. Freeman, Phillip Isola, Amir Globerson, Michal Irani, and Inbar Mosseri. Explaining in style: Training a gan to explain a classifier in stylespace. In *ICCV*, 2021.

[27] Xihui Liu, Dong Huk Park, Samaneh Azadi, Gong Zhang, Arman Chopikyan, Yuxiao Hu, Humphrey Shi, Anna Rohrbach, and Trevor Darrell. More control for free! image synthesis with semantic diffusion guidance. In *WACV*, 2023.

[28] Ze Liu, Fand Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[29] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *CVPR*, 2022.

[30] Arnaud Looveren and Janis Klaise. Interpretable counterfactual explanations guided by prototypes. In *ECML*, 2021.

[31] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *NeurIPS*, 2017.

[32] Calvin Luo. Understanding diffusion models: A unified perspective. *arXiv preprint, arXiv:2208.11970*, 2022.

[33] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 1995.

[34] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 2019.

[35] Ramaravind K. Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *FAccT*, 2020.

[36] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *ICML*, 2021.

[37] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. In *ICML*, 2022.

[38] Nick Pawlowski, Daniel Coelho de Castro, and Ben Glocker. Deep structural causal models for tractable counterfactual inference. In *NeurIPS*, 2020.

[39] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *KDD*, 2016.

[40] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

[41] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015. License: No license specified.

[42] Pouya Samangouei, Ardavan Saeedi, Liam Nakagawa, and Nathan Silberman. Explaingan: Model explanation via decision boundary crossing transformations. In *ECCV*, 2018.

[43] Shibani Santurkar, Dimitris Tsipras, Brandon Tran, Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Image synthesis with a single (robust) classifier. In *NeurIPS*, 2019.

[44] Lisa Schut, Oscar Key, Rory McGrath, Luca Costabello, Bogdan Sacaleanu, Medb Corcoran, and Yarin Gal. Generating interpretable counterfactual explanations by implicit minimisation of epistemic and aleatoric uncertainties. In *AISTATS*, 2021.

[45] Kathryn Schutte, Olivier Moindrot, Paul Hérent, Jean-Baptiste Schiratti, and Simon Jégou. Using stylegan for visual interpretability of deep learning models on medical images. In *NeurIPS Workshop*, 2020.

[46] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *IJCV*, 2019.

[47] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *ICLR*, 2014.

[48] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015.

[49] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

[50] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

[51] Suraj Srinivas and François Fleuret. Full-gradient representation for neural network visualization. In *NeurIPS*, 2019.

[52] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

[53] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019.

[54] Sahil Verma, John P. Dickerson, and Keegan Hines. Counterfactual explanations for machine learning: A review. *arXiv preprint, arXiv:2010.10596*, 2020.

[55] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harvard Journal of Law & Technology*, 2018.

[56] Zifan Wang, Haofan Wang, Shakul Ramkumar, Matt Fredrikson, Piotr Mardziel, and Anupam Datta. Smoothed geometry for robust attribution. In *NeurIPS*, 2020.

[57] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. Self-training with noisy student improves imagenet classification. In *CVPR*, 2020.

[58] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Sec. 5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Sec. 5.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A] We didn't train models.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] We have conducted qualitative experiments in the main paper for the same seed and show the diversity of VCEs across seeds in App. B.3.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See App. D
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes] Yes, directly in the citation, when applicable.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We include our code in the supplemental material.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] ImangeNet and ImageNet21k are public datasets.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [Yes] We include the results from the user study together with the screenshots of instructions in App. C.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] Not applicable.
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [Yes] See App. C.