Diffusion Visual Counterfactual Explanations

Anonymous Author(s)
Affiliation
Address
email

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 restricted 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: i) an adaptive parameterization allows us to have hyperparameters that generalize across images and models and together with distance regularization and late start of the diffusion process allow us to generate close images and ii) 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.

1 Introduction

It can be argued that one of the main problems hindering the widespread use of machine learning 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 be applied to any 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, 19, 57, 52], Shapley additive explanations [33], and local fits of interpretable models [40] have been proposed.

Moreover, [56] 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 [35] using counterfactual reasoning: "I would have recognized it as zebra (instead of horse), if it would have had black and white stripes." For a given classifier almost all methods for the generation of CEs [56, 16, 36, 6, 39, 55, 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.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

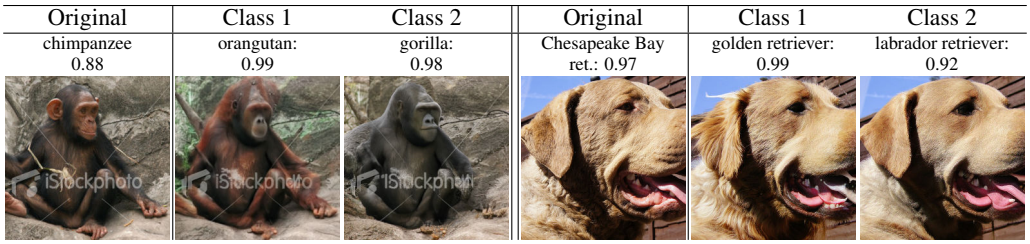


Figure 1: Our method DVCE creates visual counterfactual explanations (VCEs) that explain any given classifier (here: non-robust ConvNeXt) by changing an image of one class into a similar target class by changing class relevant features in the foreground while preserving the overall image structure.

The main problem why visual counterfactual explanations (VCEs), that is CEs for image classification, are not widely used is that the problem of generating CEs is very related to the problem of adversarial examples [53] and thus even imperceivable changes of the image can already change the prediction, which clearly does not show to 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 [21, 22, 42, 8, 20, 45, 29]. However, these approaches are either restricted to datasets with a small number of classes, cannot provide explanations for arbitrary classifiers, or the generated images look realistic but are far away from the original one, so that the VCE and the original image have so little in common that not much insight can be gained.

Recently, [28] trained a StyleGAN 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, [27] have proposed a loss to do VCEs in the latent space of a GAN/VAE. While they show promising results for MNIST/FMNIST, there is no code available for generating VCEs on ImageNet. Furthermore, they rely on a conditional GAN, which might introduce a bias as it interferes with the classifier, and it is unclear, how well an unconditional generative model will work.

In this paper, we overcome this by generating our Diffusion Visual Counterfactual Explanations (DVCEs) for arbitrary ImageNet classifiers. We use the progress in the generation of realistic images using diffusion processes [48, 26, 49, 51] reaching recently the level of GANs [15]. Similarly to [38, 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 and a user study in App. C we show that our DVCEs outperform the recent method of [7] as well as an adaptation of [2] for the generation of VCEs.

2 Diffusion models

Diffusion models [15, 48, 49, 26, 37] are generative models that consist of two steps: *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 [51].
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 Markovian 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:

$$q(x_t|x_0) = \sqrt{\overline{\alpha}_t}x_0 + \epsilon\sqrt{1-\overline{\alpha}_t}, \epsilon \sim \mathcal{N}(0, I), \tag{2}$$

where $\overline{\alpha}_t := \prod_{k=1}^{t}(1-\beta_k)$. As the number of total time steps, T approaches infinity, and the reverse transitions $q(x_{t-1}|x_t)$ approach diagonal Gaussian distributions. Then one uses a DNN to predict mean μ_θ and covariance Σ_θ for this reverse process with transition probabilities:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(\mu_\theta(x_t, t), \Sigma_\theta(x_t, t)). \tag{3}$$

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 $p_\theta(x_{t-1}|x_t)$. Originally, diffusion models were trained by optimizing a variational lower bound of the data likelihood [48]. Later, [26] proposed a simplified loss L_{simple} that resembles a denoising auto-encoder:

$$L_{\text{simple}}(\theta, T, p_{\text{data}}) := \sum_{t=1}^{T} \mathbb{E}_{x_0 \sim p_{\text{data}}} \mathbb{E}_{\epsilon \sim \mathcal{N}(0,I)} ||\epsilon - \epsilon_\theta(x_t, t)||_2^2, \tag{4}$$

where μ can be recovered as: $\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t))$. [37, 15] combined both losses to learn Σ and μ. In the continuous-time setting, [50, 25] have derived the respective lower bounds.

2.1 Class conditional sampling

There exist diffusion models trained with and without knowledge about classes in the dataset [15, 24]. For our experiments, we only use the class-unconditional diffusion model, such that all class-conditional features are introduced by the classifier which we want to explain. For the noise-aware classifier, the reverse process transitions are then of the form (Proof Appendix H in [15]):

$$p_\theta(x_{t-1}|x_t, y) = Z\, p_\theta(x_{t-1}|x_t)\, p_\theta(y|x_t, t) \tag{5}$$

for a normalization constant Z. Typically $p_\theta(y|\cdot)$ is a time-step aware classifier specifically trained on noisy images belonging to time-step t [15]. As further we would like to explain any classifier and not only noise-aware ones, we follow the approach of [2], where a classifier $p_\theta(y|\cdot)$ is given as an input the denoised sample $\hat{x}_0 = f_{\text{dn}}(x_t)$ of x_t, using the mapping:

$$f_{\text{dn}} : \mathbb{R}^d \times \{1, ..., T\} \to \mathbb{R}^d, \quad (x_t, t) \mapsto \frac{x_t}{\sqrt{\overline{\alpha}_t}} - \frac{\sqrt{1-\overline{\alpha}_t}\epsilon_\theta(x_t, t)}{\sqrt{\overline{\alpha}_t}}. \tag{6}$$

This mapping is derived from Eq. 2, from which we have that

$$x_0 = \frac{x_t}{\sqrt{\overline{\alpha}_t}} - \frac{\sqrt{1-\overline{\alpha}_t}\epsilon}{\sqrt{\overline{\alpha}_t}}, \tag{7}$$

and because we use our diffusion model $\epsilon_\theta(x_t, t)$ to predict ϵ. With this, we can define a time-step aware posterior $p_\theta(y|x_t, t)$ for any classifier $p_\theta(y|\cdot)$ as $p_\theta(y|x_t, t) = p_\theta(y|f_{\text{dn}}(x_t, t))$.

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 $p_\theta(x_{t-1}|x_t, y)$ thousands 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 [15], they proposed to solve it by approximating $p_\theta(x_{t-1}|x_t, y)$ with slightly shifted versions of $p_\theta(x_{t-1}|x_t)$ to make closed-form sampling possible. Such transition kernels are given by:

$$p_\theta(x_{t-1}|x_t, y) = \mathcal{N}(\mu_t, \Sigma_\theta(x_t, t)), \tag{8}$$

where

$$\mu_t = \mu_\theta(x_t, t) + \Sigma_\theta(x_t, t)\nabla_{x_t} \log p(y|x_t). \tag{9}$$

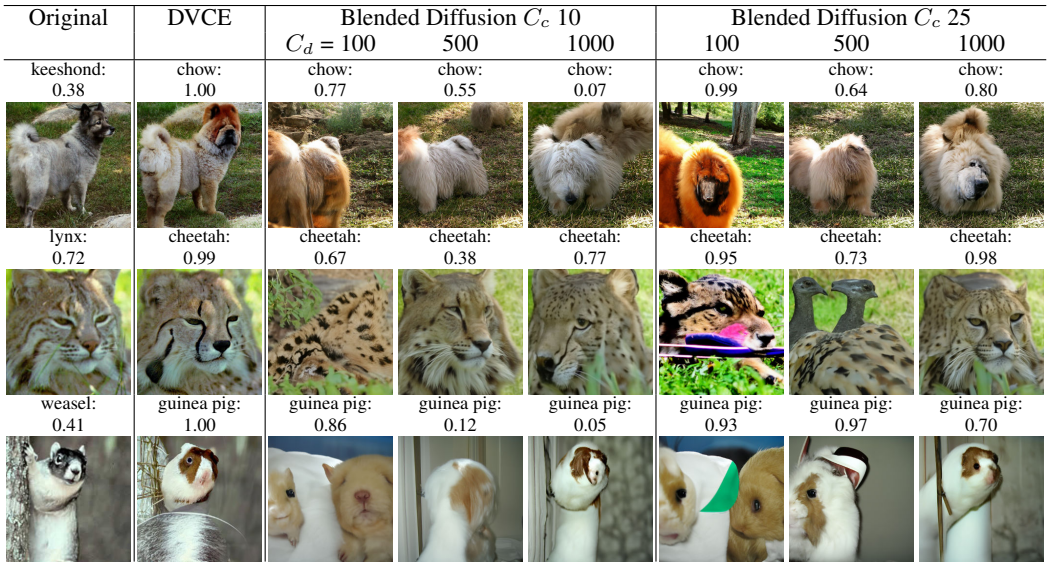


Figure 2: DVCEs for the robust Madry [17] + FT model (taken from [7]) in the second column, and DVCEs using the setting of [2], such that regularization is applied for the whole image. We can see that due to our adaptive parameterization, the same parameter works across different images for DVCEs ($C_c = 0.1, C_d = 0.15$), and across different classifiers (see Fig. 5). For blended diffusion, it is difficult to choose a set of hyperparameters that, even for a single classifier, works for multiple images. For the l_2 and LPIPS regularization weights of blended diffusion, we report the parameter of LPIPS and always use an l_2 weight 10 times as large (the ratio used in [2]).

3 Diffusion Visual Counterfactual Explanations

A VCE z for a chosen target class c and a given classifier f and input x should satisfy the following criteria: i) **validity:** the VCE z should be classified by f as the desired target class c with high probability, ii) **realism:** the VCE z should be indistinguishable or at least close to a natural image, iii) **minimality/closeness:** the difference of the VCE z and the original image x should be the minimal semantic modification necessary to change the class, in particular, the generated image z should be close to x while being valid and realistic, e.g. by changing the object in the image and leaving the background unchanged. Note that while targeted adversarial examples are valid, they do not show meaningful semantic changes into the target class for a non-robust model and are also not realistic. Although adversarial l_∞ perturbations are (almost) imperceivable for humans, adversarial images are very unlikely to occur in nature.

The $l_{1.5}$-SVCE method of [7] optimizes the predicted probability of classifier into the target class by manipulating the image and is very close to the generation of adversarial examples and thus works only for adversarially robust classifiers. The realism of the image 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 generated images are more realistic due to the integration of diffusion models. An approach similar to our DVCE framework is blended diffusion of [2] which manipulates the image inside a masked region. However, it is easy to adapt it to the task of VCEs by allowing the whole image to be manipulated. DVCE and blended diffusion share the same diffusion model, but, as we show in the Appendix, blended diffusion cannot be applied to arbitrary classifiers and requires image-specific hyperparameter tuning, see Fig. 2.

3.1 Adaptive Parameterization

For the DVCE of the original image $\hat{x}$, we don't 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. To do this, we have to condition our diffusion process on $\hat{x}$. Using the introduced denoising function f_{dn} and analogously to the derivation

in [15] and following application in [29], the mean of the transition kernel becomes:

$$\mu_\theta(x_t, t) + \Sigma_\theta(x_t, t)\nabla_{x_t}\Big[C_c \log p\big(y|f_{\text{dn}}(x_t, t)\big) - C_d d\big(\hat{x}, f_{\text{dn}}(x_t, t)\big)\Big], \tag{10}$$

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 $\exp(-\frac{d_t(\hat{x},\cdot)}{\sigma})$, where $\sigma = \frac{1}{C_d}$ and $d_t(\hat{x}, \cdot) = d(\hat{x}, f_{\text{dn}}(\cdot, t))$, thus we introduce a time-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 sparser changes. Several works have tried to minimize some distance during the diffusion process, implicitly in [9] or explicitly, but for the background in [2]. 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 blended diffusion [2] that works across images and classifiers. Thus, we propose an adaptive parametrization

$$g_{\text{update}} = C_c \frac{\nabla_{x_t} \log p_\theta(y|f_{\text{dn}}(x_t, t)))}{\|\nabla_{x_t} \log p_\theta(y|f_{\text{dn}}(x_t, t)))\|_2} - C_d \frac{\nabla_{x_t} d(\hat{x}, f_{\text{dn}}(x_t, t))}{\|\nabla_{x_t} d(\hat{x}, f_{\text{dn}}(x_t, t))\|_2}, \tag{11}$$

relative to the predicted mean of the diffusion model that we use to change μ_t in Eq. 9 to

$$\mu_t = \mu_\theta(x_{t+1}, t+1) + \Sigma_\theta(x_{t+1}, t+1)\, \|\mu_\theta(x_{t+1}, t+1)\|_2\, g_{\text{update}}. \tag{12}$$

The 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 images and even classifiers which facilitates the generation of DVCEs as otherwise hyperparameter finetuning would be necessary for each image as in blended diffusion, 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 in 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 major component of our DVCEs is that they can be applied to any image classifier regardless of whether it is adversarially robust or not. However, it turns out that naively applying diffusion with the new mean (Eq. 12) to a non-robust classifier does not work and leads to very small modifications of the image, which are similar to adversarial examples without meaningful changes. The reason is that gradients of non-robust classifiers are much noisier and less semantically meaningful. We illustrate this for a non-robust Swin-Transformer[30] in Fig. 3 where hardly any changes are generated. Note that this is even the case if we denoise the sample first using the denoising function f_{dn} (Eq. 6).

As a solution, we suggest projecting the gradient of a robust classifier, $\nabla_{x_t} \log p_{\text{robust},\theta}(y|f_{\text{dn}}(x_t, t))$, onto a cone centered at the gradient of the classifier, $\nabla_{x_t} \log p_\theta(y|f_{\text{dn}}(x_t, t))$, (we use $\alpha = 30°$ in all experiments for non-robust classifiers). More precisely, we define

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

where $\text{cone}(\alpha, v) := \{w \in \mathbb{R}^d : \angle(v, w) \leq \alpha\}$ 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 we define $u = \sin(\alpha)\frac{P_{v^\perp}(w)}{\|P_{v^\perp}(w)\|_2} + \cos(\alpha)\frac{v}{\|v\|_2}$ (note that $\|u\|_2 = 1$ and $\langle u, v\rangle = \cos(\alpha)$). The idea of this projection is that in this way we reduce drastically the noise in the gradient of the non-robust classifier. Note, however, that the projection of the gradient of the robust classifier onto the cone generated by the non-robust classifier is always an ascent direction for $\log p_\theta(y|f_{\text{dn}}(x_t, t))$ and we still have the goal of maximizing this objective (note that g_{proj} is not necessarily an ascent direction for $\log p_{\text{robust},\theta}(y|f_{\text{dn}}(x_t, t))$). Thus, one can see the cone projection as a form of regularization of $\nabla_{x_t} \log p_\theta(y|f_{\text{dn}}(x_t, t))$ which guides the diffusion process into semantically meaningful changes of the classifier.

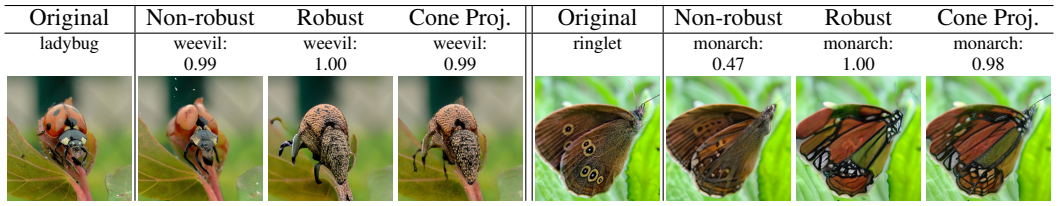


Figure 3: DVCEs for the non-robust Swin-Transformer[30], robust Madry[17] + FT model, and DVCEs for the same non-robust model. While the adversarially robust classifier is guiding the diffusion process to produce class-specific features, a non-robust classifier does not yield perceptually aligned gradients. By using a cone projection of the robust and non-robust gradients, we can satisfy all desired properties of VCEs.

3.3 Final Scheme for Diffusion Visual Counterfactuals

We summarize here the full scheme of DVCEs for a non-adversarially robust classifier $p_\theta(y|x)$ where we assume that we are given additionally the adversarially robust model $p_{\text{robust},\theta}(y|x)$:

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

For a robust model, the only difference is that the cone projection is omitted and one uses $g_{\text{proj}} = \nabla_{x_t} \log p_{\text{robust},\theta}\big(y|f_{\text{dn}}(x_t,t)\big)$. We use in all our experiments $C_c = 0.1$, and $C_d \in 0.15$ unless we show ablations for one of the parameters. The angle α for the cone projection is fixed to $30°$. In strong contrast to blended diffusion, these parameters generalize across images and even classifiers and allow us to generate VCEs for arbitrary classifiers for which a diffusion model is available.

4 Experiments

In this section, we evaluate the quality of the visual counterfactual explanations of DVCE. We start by comparing DVCE to existing works in Sec. 4.1. In Sec. 4.2, we compare counterfactuals of DVCE for various state-of-the-art ImageNet models and show, how our method can be used to interpret differences between classifiers. For all experiments, we use the same set of hyperparameters for DVCE given in the last section. The diffusion model used for DVCE is taken from [15] and is trained class-unconditionally on ImageNet images of size 256x256 and uses a modified UNet architecture (more details in [15]). User study, further experiments, ablations, and details are in the Appendix.

4.1 Method comparison

Creating VCEs for ImageNet with 1000 vastly different classes is a challenging task and there only exist a small number of methods that try to solve it without reducing dataset size or image resolution. For evaluation purposes, we compare our method to blended diffusion [2] and a recent method for VCEs based purely on optimization [7]. As the latter only works for adversarially robust classifiers, we use the multiple-norm robust model from [7], which they call Madry [17] + FT, as the classifier to create VCEs for all three methods. Results can be found in Fig. 4, where we transform one image into two different classes that are close in the WordNet hierarchy[34].

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 0.90 per image - if this is not reached, we show the one with highest confidence. For blended diffusion, 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, we again show the one with highest confidence. For DVCEs, we only use the single set of parameters provided in the previous section. As the model is robust on its own, we do not use the cone projection in DVCE.

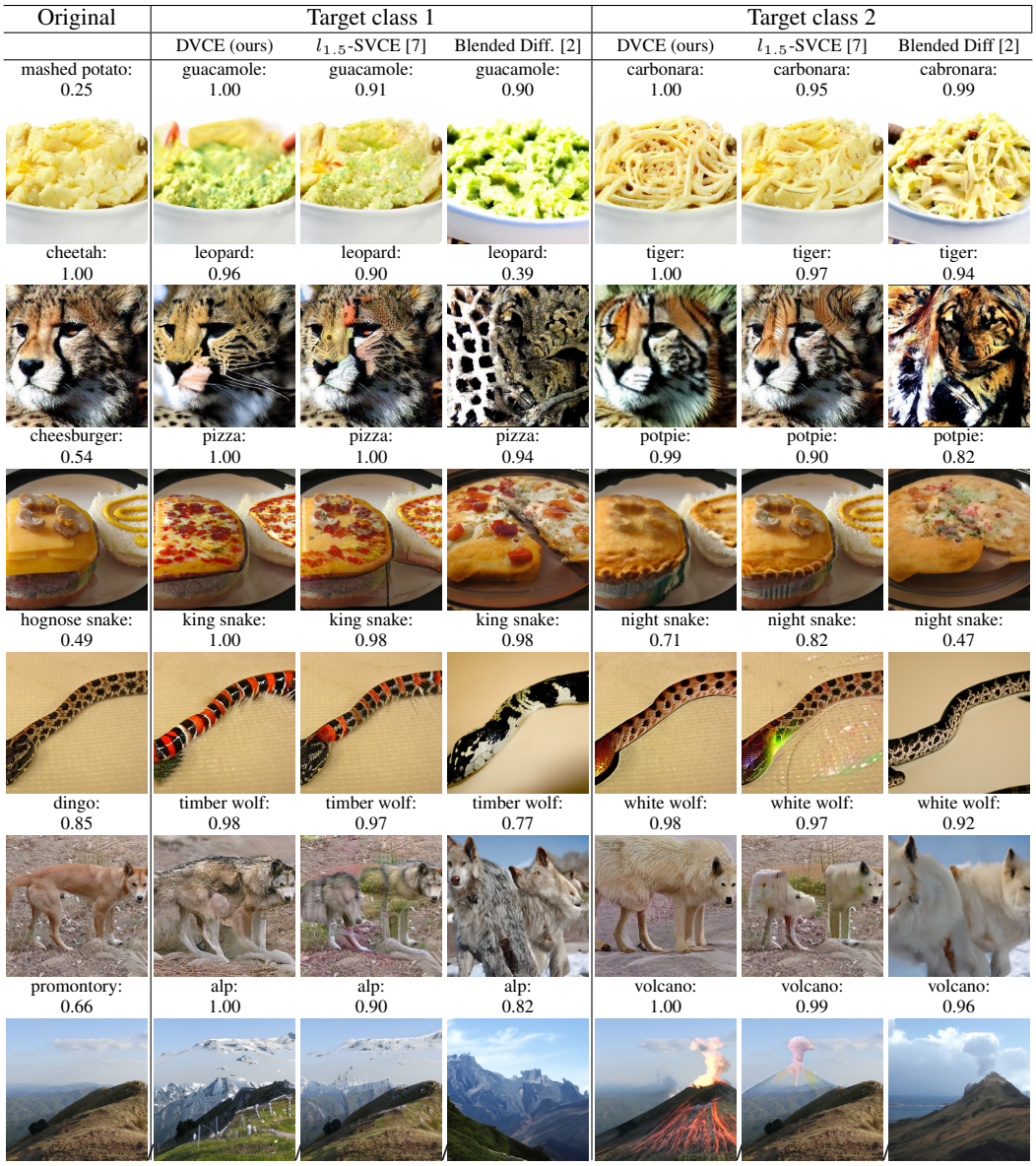


Figure 4: Comparison of different VCE-methods for the robust Madry[17] + FT (taken from [7]). We show our DVCEs (left column), the $l_{1.5}$-SVCEs of [7] (middle), and the adaptation of blended diffusion [2] for VCEs (right). For each original image (outer left column) we show the VCEs for two target classes. See the text for the parameter selection of all methods. Only DVCEs satisfy all desired properties of VCEs. Blended diffusion 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.

As Fig. 4 shows, DVCEs is the only method that satisfies all desired properties of VCEs: i) the produced VCEs have high confidence in the target class, ii) are close to natural images, and iii) show minimal semantic changes in the target class. 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. For example, it works for the "cheetah" which covers the entire image, but also for the "snake" which only

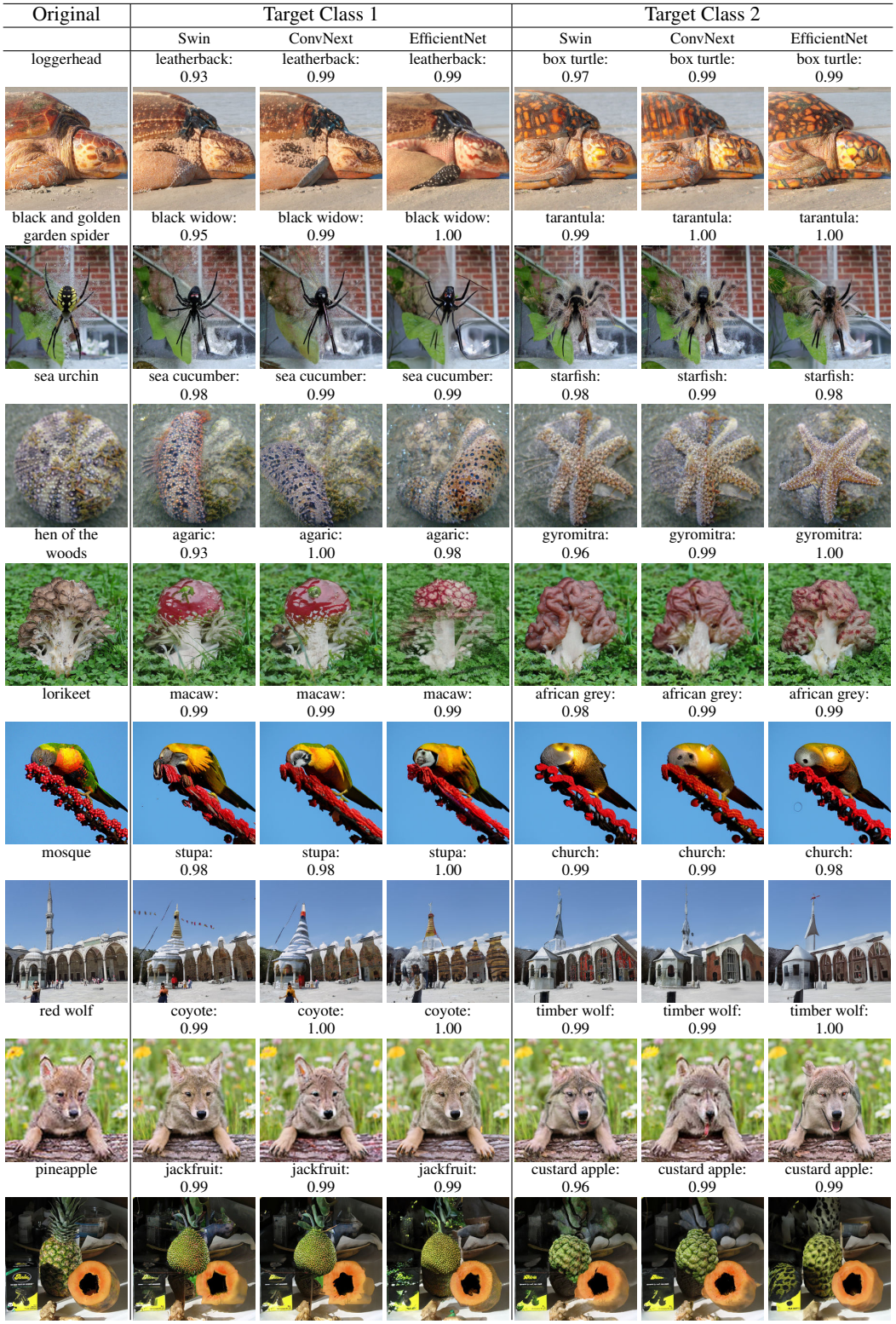


Figure 5: DVCEs for three non-robust classifiers (with ImageNet accuracy): Swin-T Transformer [30] (86.31%), ConvNext (87.02%) [31] and EfficientNet[58] (86.83%). For each original image (outer left column) we show DVCEs into two target classes. Please zoom into the images to see more fine-grained details and differences between the generated images.

covers a fraction of the input. In contrast, both $l_{1.5}$-SVCEs and blended diffusion VCEs require different hyperparameters for different images to achieve high confidence in the target class. Even with the six parameter configurations, blended diffusion is not able to produce always images with high confidence in the target class. Even more problematic for VCEs: sometimes the resulting images have high confidence but do not look realistic (cheetah $\rightarrow$ tiger) or do not 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 blended diffusion VCE 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 0.90, which often results in color artefacts and images that generally do not look realistic.

4.2 Model comparison

In Fig. 5 we show that DVCEs can be generated for various state-of-the-art ImageNet [41] models. We use a Swin-Transformer[30], a ConvNeXt [31] and a Noisy-Student EfficientNet [58, 54]. Both the Swin-Transformer and the ConvNeXt are pretrained on ImageNet21k [14] 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 visualizing arbitrary classifiers on a task as challenging as ImageNet [41]. Overall, DVCEs satisfy all desired properties of VCEs. It also allows us to inspect the most important features of each model and class. For example, all three models have learned to associate the specific texture pattern of the box turtle with that particular class. For the stupa and church classes, 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.

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 Eq. 5 is justified, in practice, however, we have to approximate the reverse transitions with shifted normal distributions of the form Eq. 8. 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 [32] 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.

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. Our DVCEs satisfy all desired properties of VCEs. Throughout the paper, we show that DVCEs can handle vastly different image configurations, object sizes, and classes.

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. *arXiv preprint, arXiv:2205.07972*, 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. *ICCV*, 2021.

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

[11] Francesco Croce and Matthias Hein. Provable robustness against all adversarial l_p-perturbations for $p \geq 1$. In *ICLR*, 2020.

[12] 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. *arXiv preprint, arXiv:2105.12508*, 2021.

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

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: a large-scale hierarchical image database. *CVPR09*, 2009. License: No license specified.

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

[16] 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.

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

[18] L. Engstrom et al. Robustness (python library), 2019.

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

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

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

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

[23] 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.

[24] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

[25] Chin-Wei Huang, Jae Hyun Lim, and Aaron Courville. A variational perspective on diffusion-based generative models and score matching. In *NeurIPS*, 2021.

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

[27] Saeed Khorram and Li Fuxin. Cycle-consistent counterfactuals by latent transformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[28] 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. *arXiv preprint, arXiv:2104.13369*, 2021.

[29] 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. *CoRR*, 2021.

[30] 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.

[31] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *arXiv preprint, arXiv:2201.03545*, 2022.

[32] Arnaud Looveren and Janis Klaise. Interpretable counterfactual explanations guided by prototypes. *arXiv preprint, arXiv:1907.02584*, 2019.

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

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

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

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

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

[38] 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. *arXiv preprint, arXiv:2112.10741*, 2021.

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

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

[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 "Medical Imaging Meets NeurIPS"*, 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. *International Journal of Computer Vision*, 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, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. In *NeurIPS*, 2021.

[51] 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.

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

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

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

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

[56] 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.

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

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

[59] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595, 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.

A Appendix

Overview of Appendix

In the following, we provide a brief overview of the additional experiments reported in the Appendix.

- In App. B, we show how varying l_1 regularization weight (B.1), the starting point of the diffusion process T (B.2), diversity (B.3), and different levels of robustness (B.4) influence the DVCEs. We add the ablation study of the angle used in the cone projection in (B.5).
- In App. C, we show the results of our user study which shows that both quantitatively and qualitatively DVCEs generate more meaningful features compared to $l_{1.5}$-SVCEs [7] and blended diffusion[2]-based VCEs.
- In App. D, we describe the hardware and resources used.
- In App. E, we add the quantitative evaluation of the realism, validity, and closeness of our DVCEs.
- In App. F, we additionally show, how our DVCEs can help to uncover spurious features.
- In App. G, we show some of the failure cases of our method.

B Ablation study of DVCEs

B.1 Regularization

In this section, we start by evaluating the impact of the distance regularization term on the diffusion process. We want VCEs to resemble the original image in the overall appearance and only change class-specific features to transfer the image into the target class. To achieve sparse changes, we use l_1-distance regularization. In Fig. 6, we vary the regularization strengths from 0.05 to 0.25. As can be seen, all regularization strengths generate meaningful target class-specific features, however, regularization C_d of 0.15 gives a good trade-off between being close to the original image as well as being realistic. For the rest of the evaluation, we, therefore, chose a weight of 0.15.

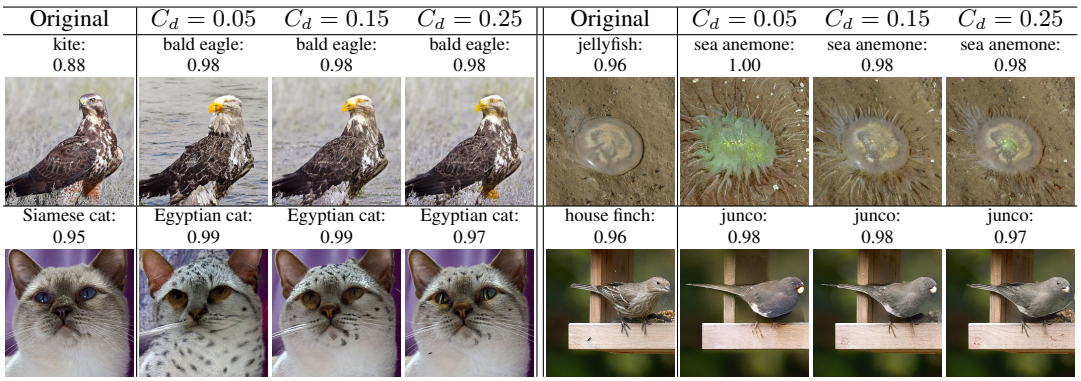


Figure 6: Ablation study for varying weights C_d for the l_1-distance term in our DVCEs for the non-robust Swin-Transformer using the cone projection introduced in Sec. 3.2.

B.2 Starting T_{start}

Next, we show how the starting time of the diffusion process influences the DVCEs. Diffusion-based sampling that is not conditioned on an original image starts the process at standard normally distributed noise. However, we show that it is easier to obtain high-quality VCEs that are similar to the original image by instead starting in the middle of the diffusion process by sampling from the closed-form probability distribution corresponding to timestep T_{start} (Eq. 2). For this, we compare three different settings and define $\eta := \frac{T_{\text{start}}}{T}$ where we choose $\eta \in \{0.25, 0.5, 0.75\}$. We observe again that $\eta = 0.5$ gives a good trade-off between closeness, realism, and showing class-specific features. Thus, in all our experiments in this paper, we chose $\eta = 0.5$.

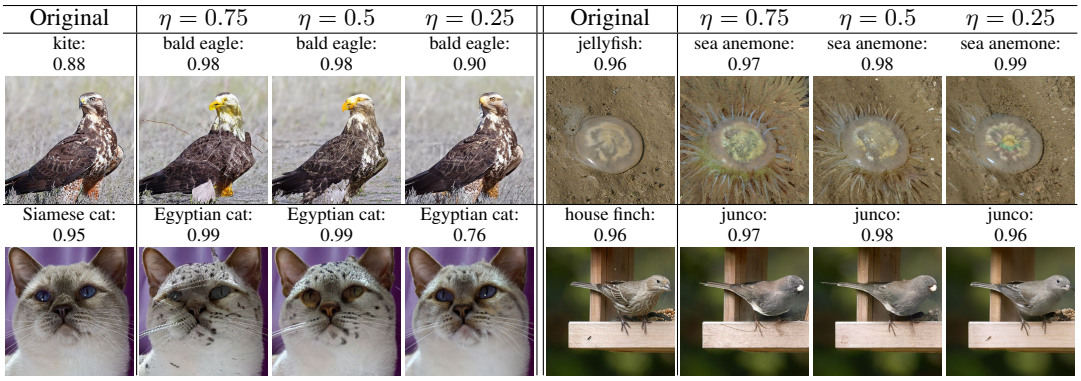


Figure 7: Comparison of different starting times $\eta := \frac{T_{\text{start}}}{T}$ of the diffusion process for our DVCEs for the non-robust Swin-Transformer using the cone projection.

B.3 Diversity

Note that the diffusion process is by design stochastic. This means that, unlike optimization-based VCEs, it is possible to generate a diverse set of VCEs from the same starting image. In Fig. 8, we visualize different DVCEs obtained for the non-robust Swin-T Transformer using the cone projection.

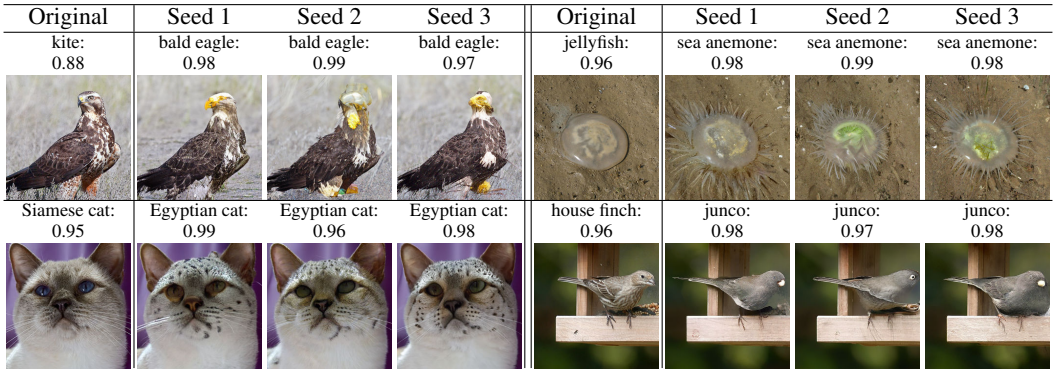


Figure 8: DVCEs for the non-robust Swin-Transformer using the cone projection across 3 different seeds for the standard parameters used in the paper. The DVCEs for different seeds show subtle variations of the generated images and satisfy the desired properties of VCEs introduced in Sec. 3.

B.4 Different robust models

Further, let us use the following notation for an l_p-ball of radius ϵ_p, $B_p(\epsilon_p) := \{x \in \mathbb{R}^d : \|x\|_p \leq \epsilon_p\}$, where d is the image dimensionality in our case.

In this section, we want to examine the influence of the choice of the robust model on DVCEs. In [12], the authors take inspiration from the result for linear classifiers in Theorem 3.1[11], which states, that when such a classifier is robust with respect to the l_1- and l_∞-balls, it is robust also in the largest l_p ball, which fits in the convex hull for any $1 \leq p \leq \infty$. This motivated them further to devise an adversarial training scheme that with some probability performs for each batch either l_1- or l_∞-attack and yields classifiers more adversarially robust for all $1 \leq p \leq \infty$. We will call this scheme multiple-norm robust finetuning.

For models trained in such a way, [7] have shown that the generated VCEs are better (both in terms of FID and qualitatively), if they are generated with respect to a sparse norm, such as $l_{1.5}$. Our DVCEs are generated with respect to an even sparser l_1 norm and thus we decide to use models with such robustness. For this, we first compare in Fig. 9 multiple-norm robustly finetuned models using the fine-tuning scheme introduced in [12]: Madry[17] + FT, XCiT[13] + FT, DeiT[5] + FT. Further, in Fig. 10, we compare cone-projection of 3 robust and 3 non-robust models.

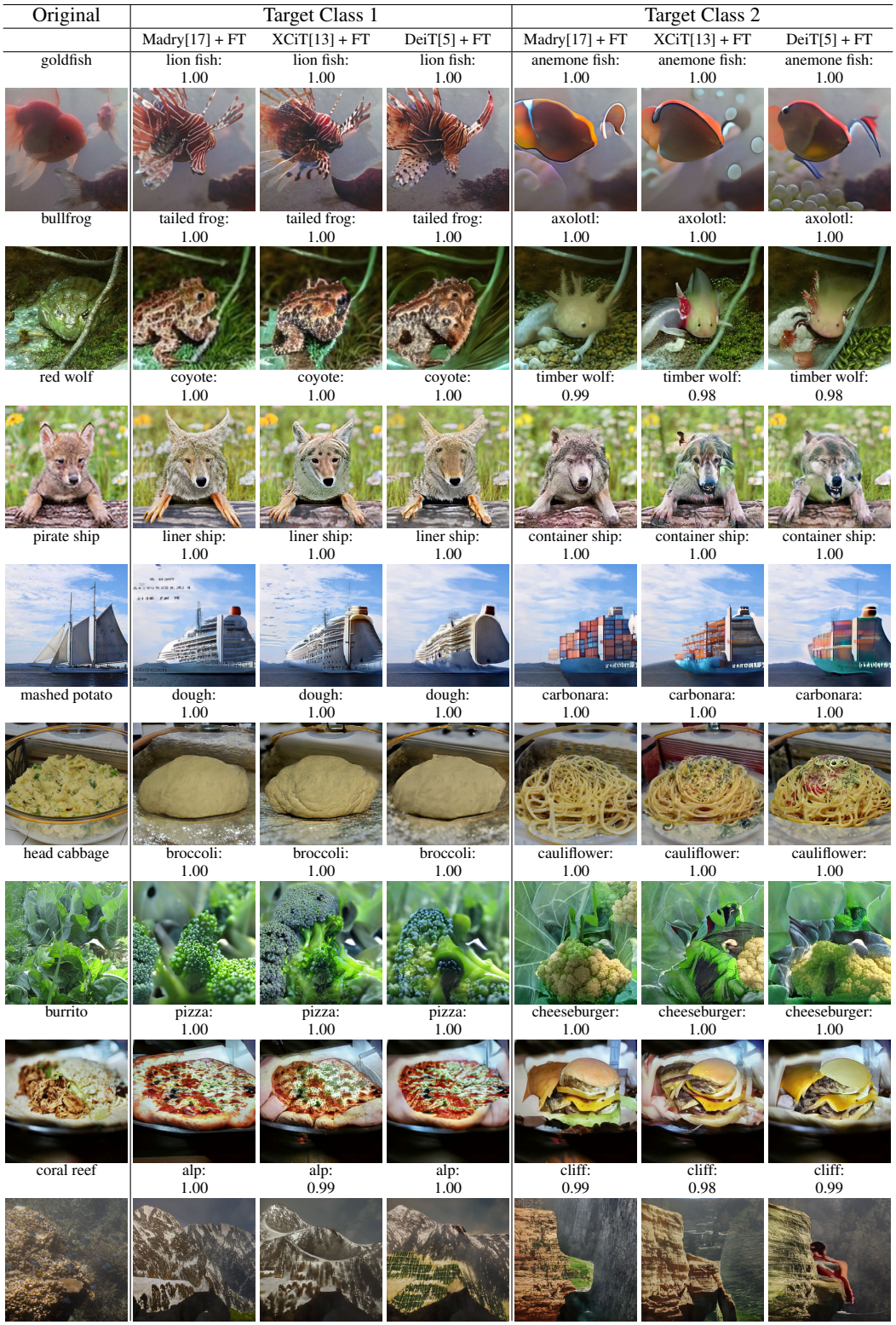


Figure 9: We compare DVCEs for three different robust models (no cone projection) which are all fine-tuned to be multiple-norm adversarially robust [12], that is against ℓ_1, ℓ_2 and ℓ_∞-perturbations.

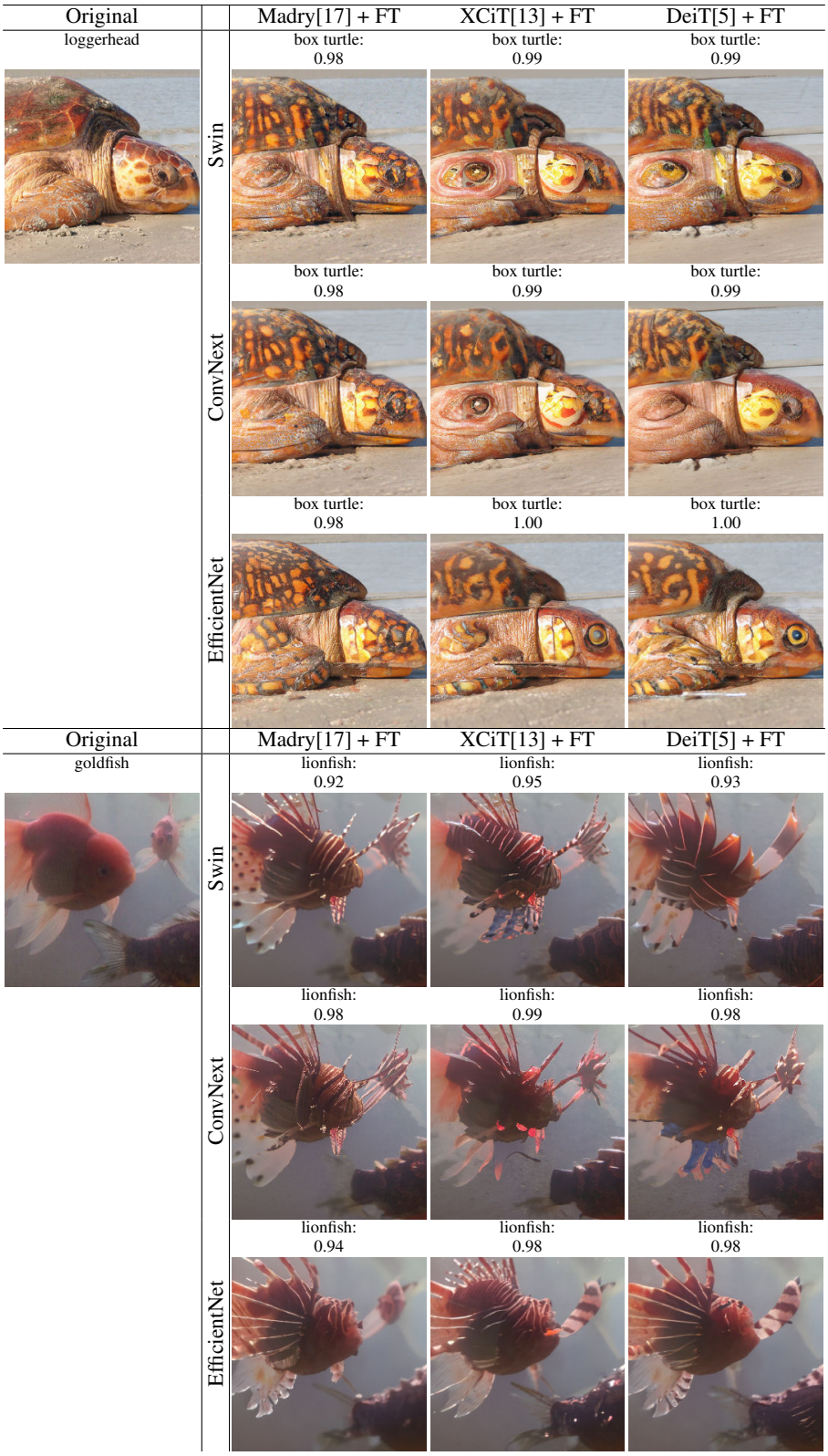


Figure 10: We compare our DVCEs for the non-robust classifiers used in Fig. 5 (rows) where we now vary additionally the adversarially robust model which is used for regularization (same as in Fig. 9 (columns)) for two different starting images. All non-robust classifiers show high confidence in the corresponding target classes and show meaningful target-specific changes. The variation of the DVCEs for the same classifier across different robust models used for the cone projection is small and on the level of the result of different seeds.

B.5 Cone projection

In this section, we are going to compare different parameters for the cone projection from Sec. 3.2, which allows us to explain non-robust classifiers. Remember that we project the gradient of the robust classifier onto a cone centered around the gradient of the target classifier. Thus, larger angles for the cone allow the method to deviate more and more from the target model gradient. In Fig. 11, we show the resulting DVCEs for various angles between 1 and $50°$. As shown in Fig. 3, the pure gradient of the target model is unable to visually guide the diffusion process, thus angles smaller than or equal to $15°$ often do not produce class-specific features in the resulting images. For angles of at least $30°$, we can see class-specific features in all images. Although it is possible to use even larger angles, we use 30° throughout the entire paper because this keeps the direction close to the target model's gradient, which is important as we want to explain the target model and not the robust one.

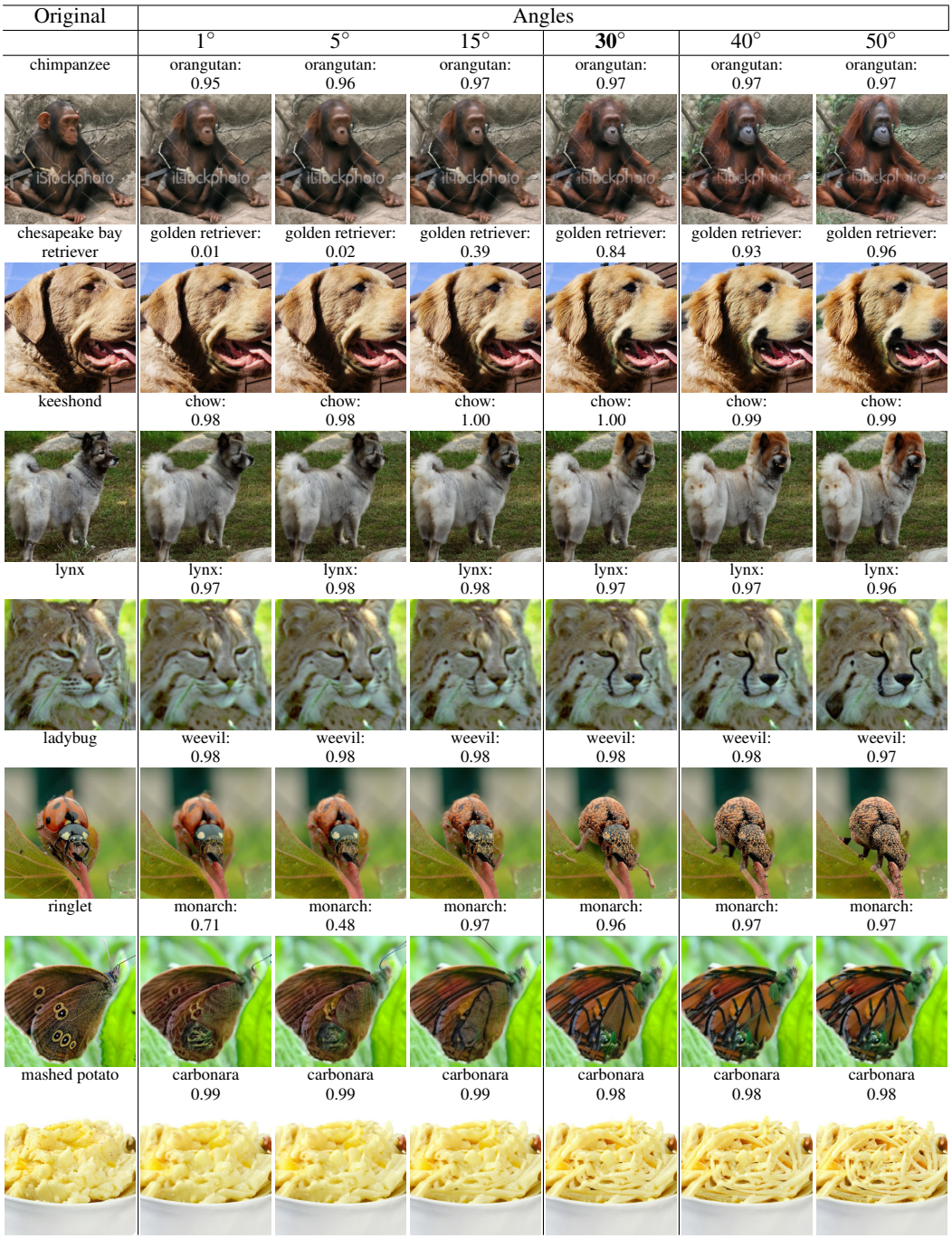


Figure 11: We compare different angles for the cone projection using the ConvNext model in combination with the Madry[18]+FT robust model. For all images one observes that class-specific features (meaningful change) and high confidence (validity) are reached for more than 30°. As the goal is to stick to the gradient of the non-robust classifier as much as possible we have fixed in the rest of the paper the angle of of the cone to 30°.

C User study

In this section, we discuss a user study that we performed to compare the $l_{1.5}$-SVCEs [7], blended diffusion-VCEs [2], and our DVCEs.

Whereas in Fig. 4 we provide a qualitative comparison showing the generated VCEs, the user study provides us also with a quantitative comparison. The images for the three methods were generated the same as for Fig. 4, see Sec. 4.1 for details. To generate the images, we randomly selected images from the ImageNet test set and then generated VCEs for two target classes (different from the original class of the image) which belong same WordNet category as the original class.

In this study, the users, who have participated voluntarily and without payment, were shown the original images and the VCEs of the three different methods in random order. The users were researchers in machine learning and related areas but none of them is working on VCEs themself or had seen the generated VCEs before. Following [7], we asked 20 users to rate 44 VCE, if the following three properties are satisfied for a given VCE (no or multiple answers are allowed): i) "Which images have meaningful features in the target class?" (**meaningful**), ii) "Which images look realistic?" **realism**, iii) "Which images show subtle, yet understandable changes?" (**subtle**). The screenshot with instructions and the shown images can be seen in Fig. 12. In the following we report

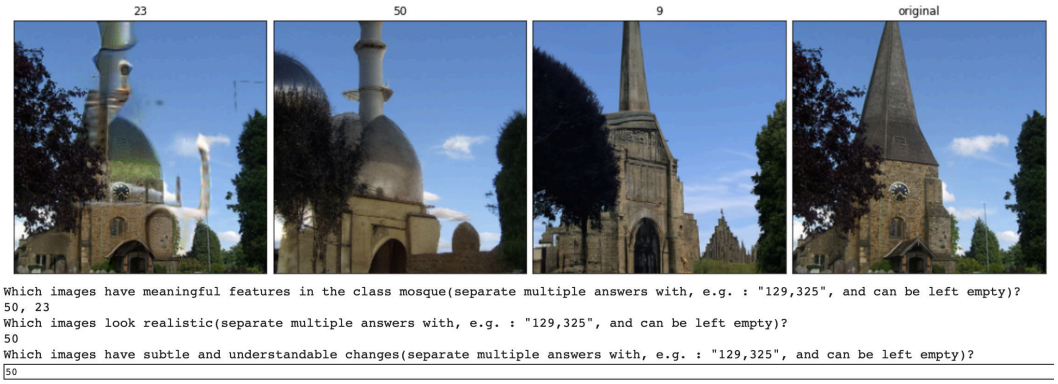


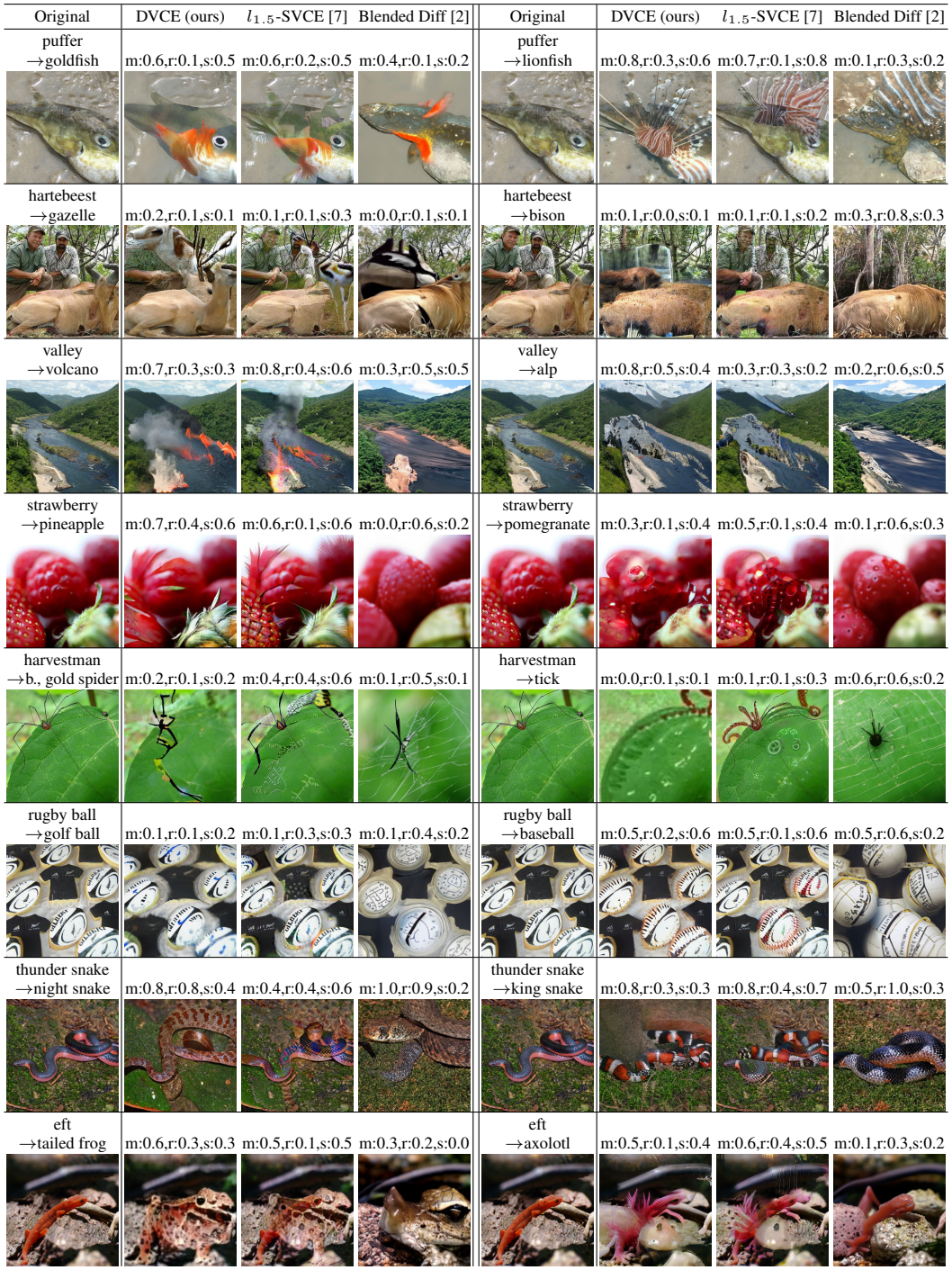
Figure 12: Screenshot of the instructions for the user study. The original image is shown on the right, then the users are shown three VCEs and are asked three questions about the changes concerning the desired target class.

the percentages of the images for DVCEs-, $l_{1.5}$-SVCEs [7], and blended diffusion [2] which were considered to have one of the three different properties: **meaningful** - **62.0%**, 48.4%, 38.7%; **realism** - 34.7%, 24.6%, **52.2%**; **subtle** - 45.0%, **50.6%**, 31.0%. In Fig. 16 we report additionally all original images together with their changes and provide the percentages for each image individually.

Our DVCEs achieve with 62.0% the highest percentage for meaningful changes, whereas blended diffusion achieves only 38.7%. However, blended diffusion VCEs are considered to be realistic in 52.2% of the cases whereas our DVCEs in 34.7%. The reason for this seemingly contradictory result is that the blended diffusion-VCE are often not able to realize a meaningful class change, in the sense that the generated images do not show the corresponding class-specific feature e.g. in Fig. 16 for the change siamang $\rightarrow$ gorilla, the blended diffusion VCE looks very good (realism 80%) but is considered to be only 20% meaningful (in fact it did not change the original image) or for jellyfish $\rightarrow$ brain coral the shown image looks realistic but shows no features of the target class. The reason is that it despite blended diffusion has the advantage that we select the image with the highest confidence from six parameter settings whereas for our method we only have one fixed parameter setting, it is sometimes not able to reach high confidence in the target class and does either too little changes (siamang $\rightarrow$ gorilla) or quite significant changes but meaningless ones (jellyfish $\rightarrow$ brain coral). Regarding $l_{1.5}$-SVCEs - they have the most subtle changes 50.6% vs. 45.0% for DVCEs but the images show often artefacts (please use zoom into images) which is the reason why they are considered the least realistic ones. In total, the user study shows that DVCE performs best among the three methods if one considers all three categories.

D Resources and hardware used

All the experiments were done on Tesla V100 GPUs, and for the generation of a batch of 6 DVCEs without cone projection and blended VCEs 3 minutes were required. For the batch of 6 DVCEs with cone projections, 5 minutes were required, as one additional model was loaded and the gradient with respect to it was calculated.



Original	DVCE (ours)	$l_{1.5}$-SVCE [7]	Blended Diff [2]	Original	DVCE (ours)	$l_{1.5}$-SVCE [7]	Blended Diff [2]
leopard →snow leopard	m:0.8,r:0.1,s:0.4	m:0.8,r:0.5,s:0.6	m:0.9,r:0.8,s:0.9	leopard →cheetah	m:0.9,r:0.2,s:0.7	m:0.4,r:0.1,s:0.2	m:0.7,r:0.7,s:0.4
dung beetle →tiger beetle	m:0.7,r:0.8,s:0.5	m:0.5,r:0.2,s:0.5	m:0.8,r:0.8,s:0.5	dung beetle →rhino. beetle	m:0.5,r:0.1,s:0.2	m:0.8,r:0.5,s:0.8	m:0.2,r:0.2,s:0.0
guinea pig →marmot	m:0.6,r:0.2,s:0.3	m:0.2,r:0.1,s:0.3	m:0.8,r:0.6,s:0.2	guinea pig →beaver	m:0.5,r:0.0,s:0.1	m:0.2,r:0.1,s:0.3	m:0.3,r:0.0,s:0.1
siamang →orangutan	m:0.7,r:0.1,s:0.4	m:0.4,r:0.1,s:0.3	m:0.6,r:0.8,s:0.6	siamang →gorilla	m:0.7,r:0.1,s:0.5	m:0.7,r:0.0,s:0.3	m:0.2,r:0.8,s:0.5
kit fox →red fox	m:0.6,r:0.1,s:0.2	m:0.3,r:0.0,s:0.2	m:0.8,r:0.7,s:0.1	kit fox →Arctic fox	m:0.5,r:0.1,s:0.2	m:0.8,r:0.5,s:0.8	m:0.0,r:0.0,s:0.0
thatch →dome	m:0.8,r:0.6,s:0.7	m:0.2,r:0.2,s:0.4	m:0.8,r:0.8,s:0.7	thatch →tile roof	m:0.8,r:0.3,s:0.6	m:0.7,r:0.1,s:0.6	m:0.9,r:0.8,s:0.8
jellyfish →sea anemone	m:0.8,r:0.5,s:0.7	m:0.5,r:0.5,s:0.6	m:0.3,r:0.4,s:0.2	jellyfish →brain coral	m:1.0,r:0.7,s:0.8	m:0.5,r:0.4,s:0.8	m:0.1,r:0.6,s:0.1

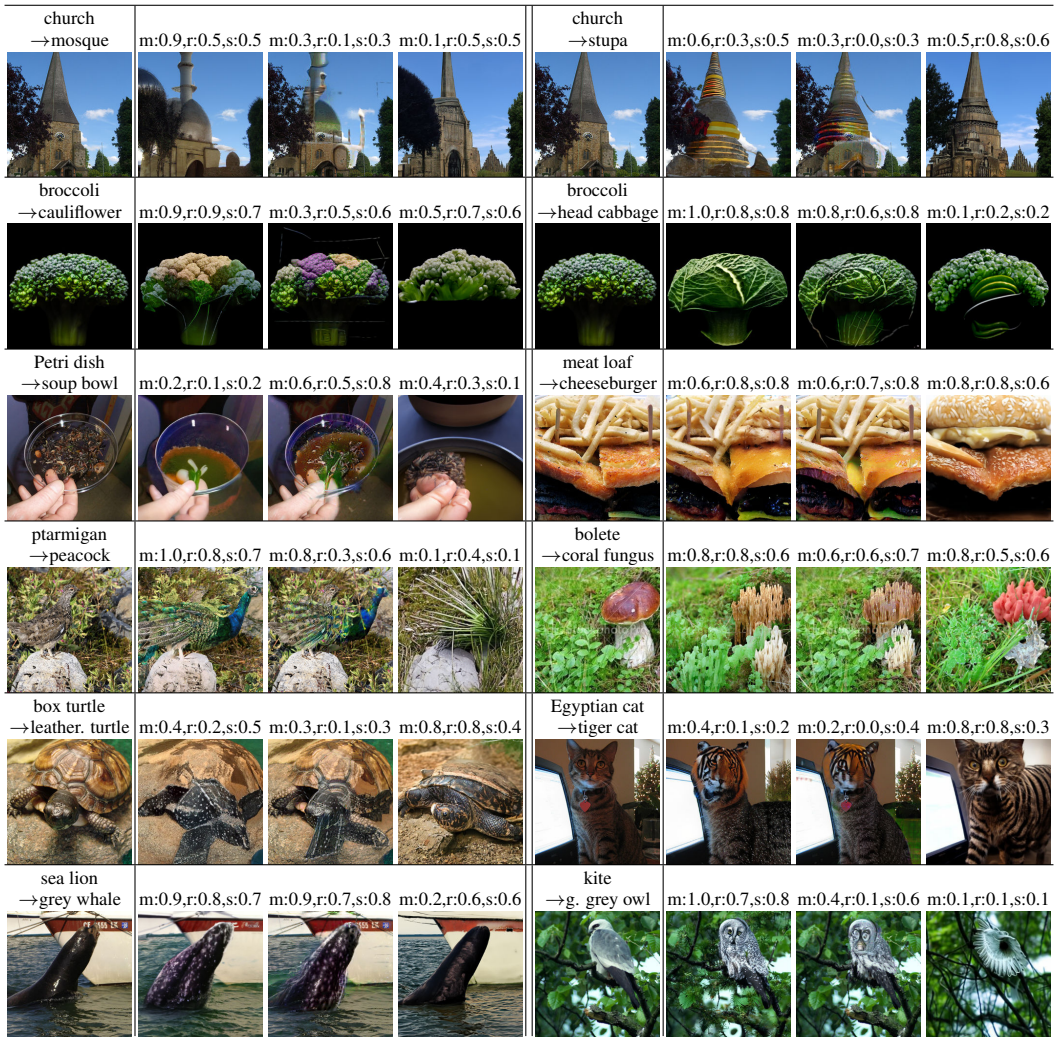


Figure 13: Comparison of different VCE-methods for the robust Madry[17] + FT (taken from [7]) for all images used for the user study from App. C. We show our DVCEs (left column), the $l_{1.5}$-SVCEs of [7] (middle), and the adaptation of blended diffusion [2] for VCEs (right) with the respective percentage of time the respective metric (introduced in App. C) - **meaningful** (**m**), **realism** (**r**), **subtle** (**s**) - is chosen by the users. Here we show both the target class and the ground truth label for all the VCEs.

E Quantitative evaluation

Here, we provide the quantitative evaluation of realism, validity, and closeness. As SVCEs [7] cannot be computed for non-robust models, we do the evaluation in this section for the robust Madry[18]+FT.

In Tab. 1, we provide a quantitative evaluation of realism, validity, and closeness of the generated VCEs.

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

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

Realism is assessed using Fréchet Inception Distance (FID) [23], which was also used in [7]. However, in [7], the authors noticed that there is a risk of having low FID scores for the VCEs that don't change the starting images significantly. To overcome this problem, we propose the following simple "crossover" evaluation:

1. divide the classes in the WordNet clusters into two disjoint sets A and B with resp. 5504 and 4576 images so that one gets a roughly balanced split of ImageNet classes.
2. compute VCEs for the test set images of classes A and B with targets (425 different classes) in B resp. A (352 different classes) (crossover). More precisely, as targets we use classes from the same WordNet cluster (see the first step) but which are in the other set. This ensures subtle changes of semantically similar classes and rules out meaningless class changes like "granny smith $\rightarrow$ container ship".
3. determine two subsets of the ImageNet trainset for the classes in A resp. B, such that in each subset the distribution of the labels corresponds to the one of A resp. B. Then we compute FID scores once between the training set corresponding to A and the VCEs generated with a target in A and for the training set corresponding to B and the VCEs with targets in B and report the average of the two FID scores.

This way, we make sure that the original images for VCEs don't come from the same distribution as the images from the subset of the training set of ImageNet. As a sanity check, we evaluated the FID scores of the training set of classes in A to the original images of the classes in B and vice versa which yields an average of 41.5. As all methods achieve a smaller FID score (lower is better), they are all able to produce features of the target classes. However, DVCE stands out with an FID score of 17.6 compared to 27.9 (blended diffusion) and 25.6 ($l_{1.5}$-SVCE).

Validity is evaluated as the mean confidence of the model achieved on the VCEs for the selected target classes. The mean confidence of DVCEs is almost the same as that of $l_{1.5}$-VCEs which maximize the confidence over the $l_{1.5}$-ball and thus are expected to have high mean confidence. So there is little difference in validity, whereas Blended diffusion has significantly lower confidence and thus has worse performance regarding validity. This is again due to the problem that the parameters leading to high confidence of the classifier but still leading to an image related to the original class are extremely difficult to find (if they exist at all) as they are image-specific.

Closeness is assessed with the set of metrics - $l_1, l_{1.5}, l_2$, as well as the perceptual LPIPS-metric [59] for the AlexNet model. As $l_{1.5}$-SVCE directly manipulate the image without the modification by a generative model, it is not surprising that in terms of l_p-distances they outperform DVCE and blended diffusion is even more apart which again is due to the problem that the images are not necessarily close to the original image.

Summary From the qualitative and the quantitative results we deduce that DVCEs and $l_{1.5}$-SVCEs are equally valid but DVCEs outperform in terms of image quality (realism) all other approaches significantly while remaining close to the original image.

F Spurious features

In this section, we want to show how DVCEs can be used as a "debugging" tool for ImageNet classifiers (resp. the training set). First, we show that DVCE finds the same spurious features as discovered in [7] and we show also more spurious features found by our method.

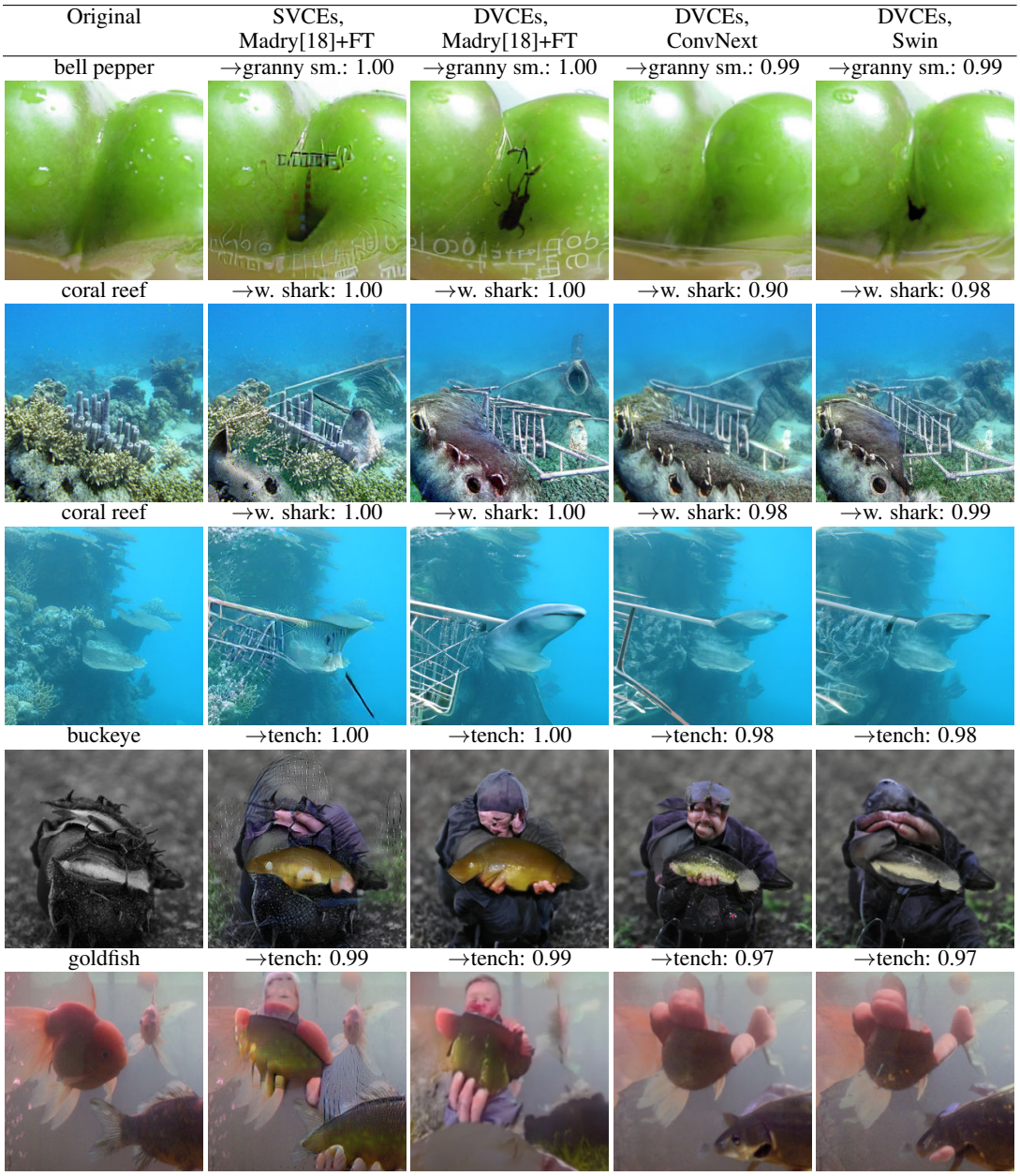


Figure 14: **Reproducing spurious features of [7] for the target classes "granny smith", "white shark", and "tench".** $l_{1.5}$-SVCEs from [7] for $\epsilon_{1.5} = 150$ and DVCEs for Madry[18]+FT. DVCEs for ConvNext and Swin have spurious features only in some cases, see last row, or second row for Swin.

Reproducing spurious features found in [7]. In Fig. 14, we first reproduce the spurious features found by [7] with $l_{1.5}$-SVCE for $\epsilon_{1.5} = 150$ for the Madry[18]+FT model. We generate the DVCE scheme for Madry[18]+FT, the non-robust Swin-Transformer [30], and ConvNeXt [31]. We find similar spurious features for the target classes "white shark" and "tench". As discussed in [7], these spurious features are a consequence of the selection of the images in the training set. For "white

shark" it contains a lot of images containing cages to protect the diver and thus the models pick up this "co-occurrence". Even more extreme for "tench" where a large fraction of images show the angler holding the tench in their hands leading to the fact that counterfactuals for tench contain human faces and hands. The spurious text feature for the class "granny smith" shown by the Madry+FT model is not visible for Swin-Transformer, and ConvNext. One key difference is that they have been trained on ImageNet-22k and then fine-tuned to ImageNet which might have reduced this spurious feature but it requires a more detailed analysis to be sure about this.

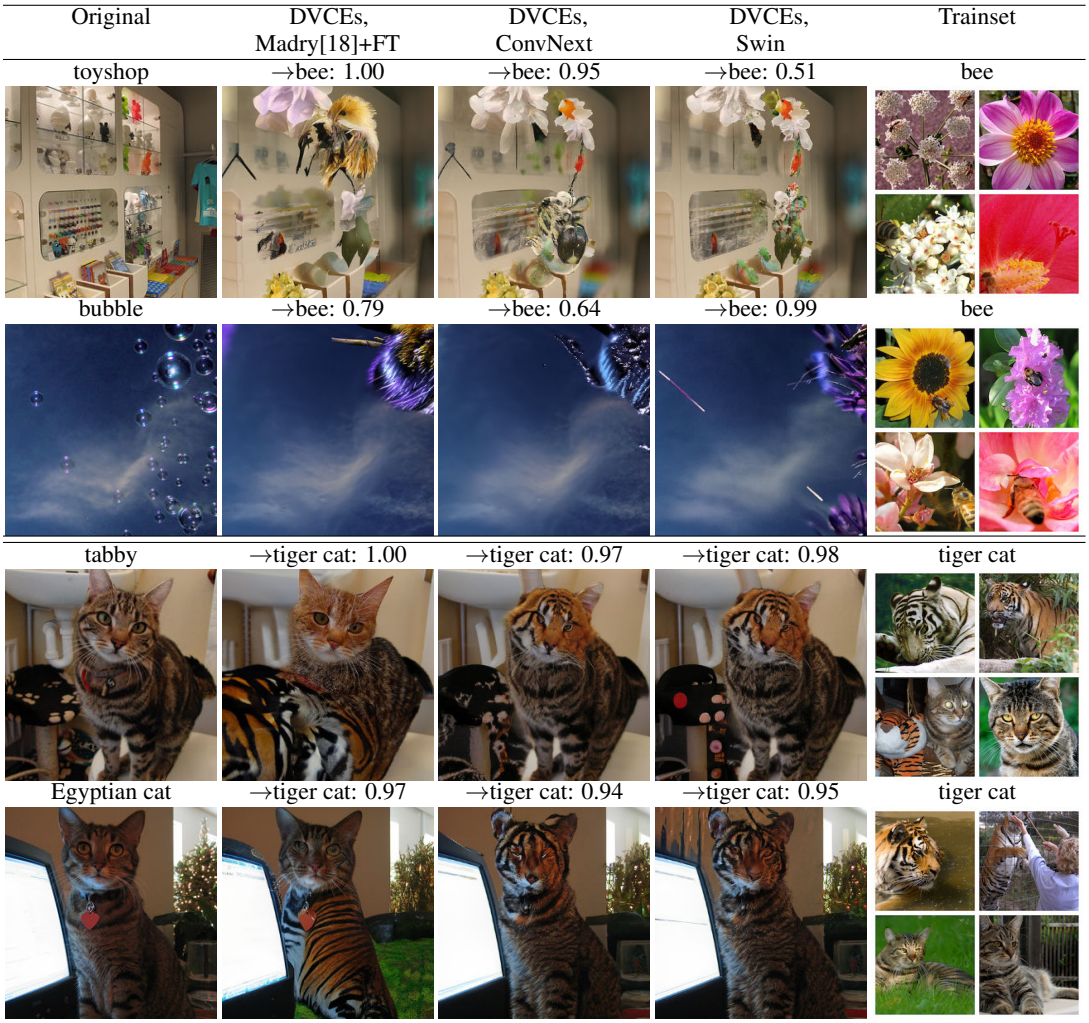


Figure 15: **New spurious features of [7] for the target classes "bee", and "tiger cat".** DVCEs for for ConvNext and Swin have spurious features for the respective target classes: flowers (and not the bee itself) for "bee", and tiger face for the "tiger cat". Here, we show both the target class and the ground truth label for all the DVCEs.

New spurious features for non-robust models. We use our DVCEs here to find novel spurious features picked up by all classifiers (but to a mixed extent) Madry[18]+FT, ConvNeXt and Swin-Transformer: features of flowers (and not the bee itself) increase the confidence in the target class "bee" and features of the tiger face increase the confidence in the target class "tiger cat" as can be seen in Fig. 15. Here, we show both the target class and the ground truth label for all the VCEs. We also show in the rightmost column the samples from the trainset of ImageNet from the respective target classes. In both cases, they show why the classifier has picked up these spurious features. Images of bees often show flowers, most of the time much larger than the bee itself. Whereas the DVCE for "tiger cat" shows that the training set of this class is completely broken as it contains images of "tigers" (note that "tiger" is a separate class in ImageNet).

G Failure cases

From what we have seen, generally, there are two failure cases of our method: i) sometimes DVCEs can be blurry, which seems to be an artefact of the diffusion model as it happens for blended diffusion and is visible also in the original diffusion paper (see top left image in Fig. 15 of [15]), ii) DVCEs can fail when the change is difficult to realize (i.e. the original image is not similar to the target class).

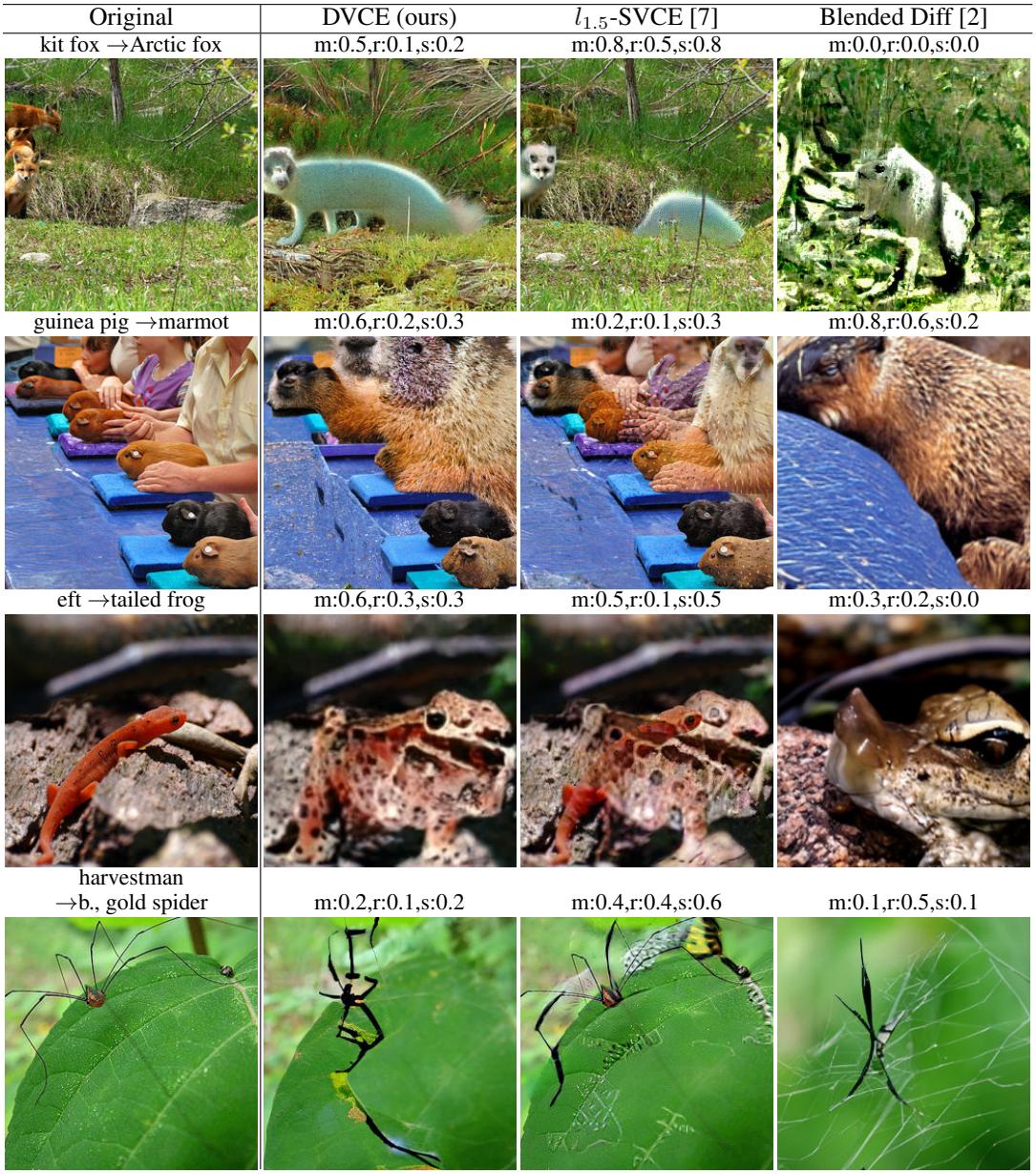


Figure 16: **Some failure cases of DVCEs for the robust Madry[17]+FT from our user study in App. C**. The failure cases that we have seen are: i) DVCEs (and blended diffusion) can be slightly blurry, or ii) change is difficult to realize. As in Fig. 16, we show our DVCEs (left column), the $l_{1.5}$-SVCEs of [7] (middle), and the adaptation of blended diffusion [2] for VCEs (right) with the respective percentage of time the respective metric (introduced in App. C) - **meaningful** (**m**), **realism** (**r**), **subtle** (**s**) - is chosen by the users. Here, we show both the target class and the ground truth label for all the VCEs.