Supplementary Material: Adversarial Counterfactual Visual Explanations

A. Detailed Implementation Details

For each dataset, we used different configurations in architecture and for the generation of the pre-explanation. Yet, we tune all hyperparameters from a empirical perspective. We tuned $\tau$ such that the input image and its filtered instance are visually similar. Additionally, the classification between these two images are the same. To adjust the hyperparameter $\lambda_d$, we performed a simple visual inspection. Finally, for the threshold, we ablated its values empirically for each dataset. When using the distance loss $\ell_1$, we set the distance regularization constant to $\lambda_d = 0.001$ while $\lambda_d = 0.1$ for $\ell_2$. For the final refinement, firstly, we normalize the mask by the maximum pixel's difference magnitude. For the dilation step, we set the mask as a square with a width and height of 15 pixels for all datasets. Finally, we used the cross entropy for all experiments as the $L_{\text{class}}$ loss. Next, we will show all implementation details for each dataset.

CelebA [5]: We used the same architecture and weights as [3]. Additionally, we set $\tau = 5$ with a total amount of steps as 50. At the refinement stage, we used the same threshold of 0.15 for both $\ell_1$ and $\ell_2$ experiments for smile and age attributes.

CelebA HQ [4]: Our model follows the same architecture than [2] for ImageNet $256 \times 256$ unconditional generation. Since CelebA HQ is far less complex than ImageNet, we reduced the number of channels from 256 to 128. Also, our model generates samples using 500 diffusion steps instead of 1000. For training, we iterated our model for 120,000 iterations with a batch size of 256 on two V100 GPUs following [2]'s code. We set the learning rate to $10^{-4}$, a weight decay of 0.05, and no dropout.

To generate the pre-explanations, we noise the image until $\tau = 5$ out of 25 re-spaced steps. To binarize the mask, we used a threshold of 0.15 and 0.1 for the smiling attribute with the $\ell_1$ and $\ell_2$ distance losses, respectively. For the age attribute, we used 0.15 for $\ell_1$ and 0.05 for $\ell_2$.

BDD100k/OIA [6, 7]: The counterfactual explanation research community opted to use BDD100k in a $512 \times 256$ setup. This is highly demanding computationally to create a DDPM. Thus, since we knew a priori that we do not need many iterations for ACE to generate counterfactuals, we trained our diffusion model partially in the Markov chain. That is, our DDPM cannot generate images from pure noise. Instead, we trained it to generate images solely from a quarter of the complete chain, requiring an input instance to warm up the generation. So, we trained our model to generate instances with 250 steps out of 1000. This enabled us to use a lighter model. Architectonically, our UNet model has four downsampling stages with 128 $s$ channels, where $s$ is the downsampling stage. Finally, we used the attention layer at the deeper layer of the UNet. At the training phase, we used a batch size of 256, a learning rate of $10^{-3}$, and a weight decay and dropout of 0.05 for 50,000 iterations.

To generate our explanations, we used 5 out of 100 (re-spaced) diffusion steps. For $\ell_1$, we used a threshold of 0.05 and 0.1 for $\ell_2$ for both datasets.

ImageNet [1]: For this dataset, we took advantage of previous works. In this case, we utilised [2]'s model on ImageNet 256. To generate the explanations, we used 5 steps out of 25 for the pre-explanations and set the threshold to 0.15 to binarize the mask for all cases.

B. Overview of ACE

ACE is a two-step method: firstly is the pre-explanation construction – Algorithm 1 – and then the refinement process – Algorithm 2. To generate the pre-explanation, (1) we add noise to the input image $x$ using the forward Markov chain until an intermediate step $\tau$, i.e. it doesn’t begin from random Gaussian noise. Instead, it warms up the generation with the input image through

$$x_{\tau} = \sqrt{\alpha_t} x + \sqrt{1 - \alpha_t} \epsilon, \epsilon \sim \mathcal{N}(0, I).$$

(2) ACE iteratively denoises the noisy image using the DDPM algorithm with

$$x_{\tau-1} = \mu_t(x_\tau) + \Sigma_t(x_\tau) \epsilon, \epsilon \sim \mathcal{N}(0, I),$$

where $\mu_t$ and $\Sigma_t$ are the output of the diffusion model.

(3) The scrutinized classifier uses the filtered image to compute loss function. Then, we calculate the gradients with

\footnotesize
Note that all these hyperparameters are not the same as the classically found in machine learning. These variables can be adjusted by the user in an ‘online’ manner according to his/her expectations. Hence, a global configuration is a mere rough estimate of these parameters and can be accommodated instance-wise.
respect to the input image \(x\) in step 1, all the way through the \(\tau\) steps of the diffusion model. (4) ACE applies the gradients as the update step with the attack of choice. It iterates these four steps to create the pre-explanation. For the refinement, it creates the mask \(m\) using the difference between the pre-explanation and the original input. Then, it dilates and thresholds it to generate the binary version. Finally, ACE builds on RePaint to keep untouched any region lying outside the mask. The final result is the counterfactual explanation.

Algorithm 1 Pre-explanation generation

**Require:** Diffusion Model \(D\), Distance loss \(d\) and its regularization constant \(\lambda_d\), classification loss \(L_{class}\) comprising the classifier under observation, number of noising steps \(\tau\), attack optimization algorithm \(PGD\), number of update iterations \(n\), initial instance \(x\), target label \(y\)

1: function PRE-EXPLANATION(x, y)
2: \(n \leftarrow 0\)
3: \(x_{orig} \leftarrow x\)
4: while \(n < N\) do ▶ Attack iteration steps
5: \(\epsilon \sim \mathcal{N}(0, I)\)
6: \(x' \leftarrow \sqrt{\alpha_\tau}x + \sqrt{1-\alpha_\tau}\epsilon\) ▶ Add noise
7: \(ts \leftarrow \tau - 1\)
8: while \(ts \geq 0\) do ▶ DDPM denoising
9: \(\mu, \Sigma \leftarrow D(x', ts)\)
10: \(\epsilon \sim \mathcal{N}(0, I)\)
11: \(x' \leftarrow \mu + \epsilon\Sigma\)
12: \(ts \leftarrow ts - 1\)
13: end while
14: \(g \leftarrow \nabla_{x'}L_{class}(x'; y') + \lambda_d(x', x_{orig})\)
15: \(x \leftarrow PGD(x, g)\) ▶ Update with attack
16: \(n + 1 \leftarrow n\)
17: end while
18: return \(x'\) ▶ Pre-explanation
19: end function

C. Qualitative Results

In this section, we show more qualitative results. We will display the input image, its pre-explanation, the mask, and the final counterfactual for both \(\ell_1\) and \(\ell_2\) losses on all datasets. Note that we added a small discussion on the caption analyzing the results. In Fig. 10, we compare a few examples of DiME and ACE.

Algorithm 2 Post-processing

**Require:** Diffusion Model \(D\), number of noising steps \(\tau\), mask dilation size \(d\), threshold \(u\), initial instance \(x\), pre-explanation \(x'\)

1: function POST-PROCESSING(x, x')
2: \(x_{orig} \leftarrow x\)
3: \(\epsilon \sim \mathcal{N}(0, I)\)
4: \(x' \leftarrow \sqrt{\alpha_\tau}x + \sqrt{1-\alpha_\tau}\epsilon\)
5: \(ts \leftarrow \tau - 1\)
6: # Mask generation
7: \(m \leftarrow sum\_over\_channels(abs(x - x'))\)
8: \(m \leftarrow m/\max(\text{maximum}(m))\)
9: \(m \leftarrow \text{dilation}(m, \text{size} = d) > u\)
10: while \(ts \geq 0\) do ▶ DDPM denoising
11: \(\epsilon \sim \mathcal{N}(0, I)\)
12: \(x_{ts} \leftarrow \sqrt{\alpha_\tau}x + \sqrt{1-\alpha_\tau}\epsilon\)
13: \(x' \leftarrow m \cdot x' + (1 - m) x_{ts}\)
14: \(\mu, \Sigma \leftarrow D(x', ts)\)
15: \(\epsilon \sim \mathcal{N}(0, I)\)
16: \(x' \leftarrow \mu + \epsilon\Sigma\)
17: \(ts \leftarrow ts - 1\)
18: end while
19: return \(x'\) ▶ Counterfactual explanation
20: end function

References


Figure 1. Additional CelebA qualitative results. We show examples for the *Smiling* attribute for both distances losses. From our qualitative experiments, we see that removing the smile attributes is harder than adding them. Additionally, we see that the $\ell_1$ loss creates more sparse editings.

Figure 2. Additional CelebA qualitative results. We show examples for the Age attribute for both distances losses. The results show that the $\ell_1$ loss creates more out-of-distribution artifacts.
Figure 3. Additional CelebA HQ qualitative results. We show examples for the Smiling attribute for both distances losses. We see similar behavior in the CelebA dataset.
Figure 4. Additional CelebA HQ qualitative results. We show examples for the Age attribute for both distances losses. These examples show that transforming Old to Young is less informative than the other way.
Figure 5. Additional BDD qualitative results. We show examples for the Forward / Slow Down binary class for $\ell_2$ distance loss. We show a zoom of the changes in the image since the perturbations are sparse. We see that ACE adds traffic light colors in the buildings to change the prediction.
Figure 6. Additional BDD qualitative results. We show examples for the Forward / Slow Down binary class for $\ell_1$ distance loss. We show a zoom of the changes in the image since the perturbations are sparse. We see that ACE adds traffic light colors in the buildings to change the prediction.
Figure 7. Additional ImageNet qualitative results. We show examples for the Zebra / Sorrel categories class. The first column is the $\ell_1$ distance loss while the second one is $\ell_2$. The initial row is zebra to sorrel and the second one is the inverse. To change from zebras to sorrels, some examples show not only incorporating the brown color sorrel horses but also the context in the background (e.g. adding a stable-like background). Vice-versa, to classify a horse as a zebra it is enough to add some strips.
Figure 8. Additional ImageNet qualitative results. We show examples for the Cheetah / Cougar categories class. The first column is the $\ell_1$ distance loss while the second one is $\ell_2$. The first row is cheetah to cougar and the second is the inverse. We mainly see that changing from cheetah to cougar is enough to target the face of the animal. Vice-versa, to classify a cougar as a cheetah, ACE adds spots and characteristic cheetah stripes on the face.
Figure 9. Additional ImageNet qualitative results. We show examples for the Egyptian / Persian cat categories class. The first column is the $\ell_1$ distance loss while the second one is $\ell_2$. The row is Egyptian to Persian cat and the second is the inverse. To change from Egyptian to Persian, we mainly see that ACE adds the Persian cats’ fluffy fur. Conversely, from Persian to Egyptian it adds spots.
Figure 10. ACE vs. DiME. We display some examples showing some differences between DiME counterfactuals and ACE’s. In short, ACE is capable of not modifying useless information, such as the background, to generate its counterfactuals. Top row: CelebA. Bottom row: CelebA HQ. Left Column: Smiling attribute. Right Column: Age attribute.