Abstract

We present a method for neural network interpretability by combining feature attribution with counterfactual explanations to generate attribution maps that highlight the most discriminative features between pairs of classes. We show that this method can be used to quantitatively evaluate the performance of feature attribution methods in an objective manner, thus preventing potential observer bias. We evaluate the proposed method on three diverse datasets, including a challenging artificial dataset and real-world biological data. We show quantitatively and qualitatively that the highlighted features are substantially more discriminative than those extracted using conventional attribution methods and argue that this type of explanation is better suited for understanding fine grained class differences as learned by a deep neural network.

1 Introduction

Machine Learning—and in particular Deep Learning—continues to see increased adoption in crucial aspects of society such as industry, science, and healthcare. As such, it impacts human lives in significant ways. Consequently, there is a need for understanding how these systems work and how they make predictions in order to increase user trust, limit the perpetuation of societal biases, ensure correct function, or even to gain scientific knowledge. However, due to the large numbers of parameters and non-linear interactions between input and output, deep neural networks (DNNs) are generally hard to interpret. In particular, it is not clear which input features influence the output of a DNN.

A popular approach for explaining DNN predictions is provided by so-called feature attribution methods. These methods output the importance of each input feature w.r.t. the output of the DNN. In the case of image classification—the primary focus of this work—the output is a heatmap over input pixels, highlighting and ranking areas of importance. A large number of approaches for feature attribution have been proposed in recent years (for a recent review see Samek et al. (2021) and related work below). Although those have been used successfully to interpret model behavior for some applications, there is still debate about the effectiveness, accuracy, and trustworthiness of these approaches (Kindermans et al., 2019; Adebayo et al., 2018; Ghorbani et al., 2019; Alvarez-Melis and Jaakkola, 2018). In addition, objectively evaluating feature attribution methods remains a difficult task (Samek et al., 2016; Hooker et al., 2018).
While counterfactual explainability methods have seen increased adoption in structured data domains, a complementary approach for explaining DNN decisions are so called counterfactual explanations (Martens and Provost, 2014; Wachter et al., 2017). In contrast to feature importance estimation, counterfactual approaches attempt to explain a DNN output by presenting the user with another input that is close to the original input, but changes the classification decision of the DNN to another class. For humans, this representation is natural and it provides an intuitive means for elucidating DNN behaviour.

While counterfactual explainability methods have seen increased adoption in structured data domains, they are comparatively less popular for image data, where feature attribution methods arguably remain the dominant tool for practitioners. The popularity of feature importance methods is partly driven by their ease of use, availability in popular Deep Learning frameworks, and intuitive outputs in the form of pixel heatmaps. In contrast, generating counterfactual explanations typically involves an optimization procedure that needs to be carefully tuned in order to obtain a counterfactual with the desired properties. This process can be computationally expensive and does, in general, not allow for easy computation of attribution maps (Verma et al., 2020).

To address these issues, we present a simple method that bridges the gap between counterfactual explainability and feature importance for image classification by building attribution maps from counterfactuals (DAC: Discriminative Attribution from Counterfactuals, see Fig. 1 for a visual summary). Crucially, our method can be used to quantitatively evaluate the attribution in an objective manner on a target task, a missing feature in current attribution methods. We use a cycle-GAN (Zhu et al., 2017) to translate real images $x_R$ of class $i$ to counterfactual images $x_C$ of class $j$, such that the classifier $f$ we wish to interpret predicts $y_R = i$ and $y_C = j$. The Discriminative Attribution method then searches for the minimal mask $m$, such that copying the most discriminative parts of the real image $x_R$ into the counterfactual $x_C$ (resulting in the hybrid $x_H$) is again classified as $y_H = i$.

A complementary approach for explaining DNN decisions are so called counterfactual explanations (Martens and Provost, 2014; Wachter et al., 2017). In contrast to feature importance estimation, counterfactual approaches attempt to explain a DNN output by presenting the user with another input that is close to the original input, but changes the classification decision of the DNN to another class. For humans, this representation is natural and it provides an intuitive means for elucidating DNN behaviour.

While counterfactual explainability methods have seen increased adoption in structured data domains, they are comparatively less popular for image data, where feature attribution methods arguably remain the dominant tool for practitioners. The popularity of feature importance methods is partly driven by their ease of use, availability in popular Deep Learning frameworks, and intuitive outputs in the form of pixel heatmaps. In contrast, generating counterfactual explanations typically involves an optimization procedure that needs to be carefully tuned in order to obtain a counterfactual with the desired properties. This process can be computationally expensive and does, in general, not allow for easy computation of attribution maps (Verma et al., 2020).

To address these issues, we present a simple method that bridges the gap between counterfactual explainability and feature importance for image classification by building attribution maps from counterfactuals (DAC: Discriminative Attribution from Counterfactuals, see Fig. 1 for a visual summary). Crucially, our method can be used to quantitatively evaluate the attribution in an objective manner on a target task, a missing feature in current attribution methods. We use a cycle-GAN (Zhu et al., 2017) to translate real images $x_R$ of class $i$ to counterfactual images $x_C$ of class $j \neq i$, where we validate that the translation has been successful by confirming that $f(x_R) = i$ and $f(x_C) = j$, where $f$ is the classifier to interpret. We repurpose a set of common attribution methods by introducing their discriminative counterparts, which are then able to derive attribution maps from the paired real and counterfactual image. We show that this approach is able to generate sparse, high quality feature attribution maps that highlight the most discriminative features in the real and counterfactual image more precisely than standard attribution methods. Furthermore, subsequent thresholding of the attribution map allows us to extract binary masks of the features and quantify their discriminatory power by performing an intervention and replacing the highlighted pixels in the counterfactual with the corresponding pixels in the real image. The difference in output classification score of this hybrid image, compared to the real image classification, then quantifies the importance of the swapped features. We validate our method on a set of three diverse tasks, including a challenging artificial dataset, a real world biological dataset (where a DNN solves a task human experts can not), and MNIST. For all three datasets we show quantitatively and qualitatively that our method outperforms all considered attribution methods in identifying key discriminatory features between the classes. Source code and datasets are publicly available at https://dac-method.github.io.

### 2 Related Work

Recent years have seen a large number of contributions addressing the problem of interpretability of DNNs. These can be broadly distinguished by the type of explanation they provide, either *local* or *global*. Methods for local interpretability provide an explanation for every input, highlighting the
reasons why a particular input is assigned to a certain class by the DNN. Global methods attempt to distill the DNN in a representation that is easier to understand for humans, such as decision trees. One can further distinguish between interpretability methods that are post-hoc, i.e., applicable to every DNN after it has been trained, and those methods that require modifications to existing architectures to perform interpretable classification as part of the model. In this work we focus on a specific class of local, post-hoc approaches to DNN interpretability for image classification, so-called feature importance estimation methods.

**Attribution Methods for Image Classification** Even in this restricted class of approaches there is a large variety of methods (Ribeiro et al., 2016; Lundberg and Lee, 2017; Baehrens et al., 2010; Bach et al., 2015; Zintgraf et al., 2017; Selvaraju et al., 2017; Sundararajan et al., 2017; Simonyan et al., 2014; Zeiler and Fergus, 2014; Kindermans et al., 2017; Montavon et al., 2017; Fong and Vedaldi, 2017; Dabkowski and Gal, 2017; Zhang et al., 2016; Shrikumar et al., 2017, 2016; Smilkov et al., 2017). They have in common that they aim to highlight the most important features that contributed to the output classification score for a particular class, generating a heatmap indicating the influence of input pixels and features on the output classification. Among those, of particular interest to the work presented here are baseline feature importance methods, which perform feature importance estimation with reference to a second input. Those methods gained popularity, as they assert sensitivity and implementation invariance (Sundararajan et al., 2017; Shrikumar et al., 2016). The baseline is usually chosen to be the zero image as it is assumed to represent a neutral input.

**Counterfactual Interpretability** Another body of literature that is relevant to the presented work are counterfactual interpretability methods first proposed by Martens and Provost (2014). Since then, the standard approach for generating counterfactuals broadly follows the procedure proposed by Wachter et al. (2017), in which the counterfactual is found as a result of an optimization aiming to maximize output differences while minimizing input differences between the real image \(x_R\) and the counterfactual \(x_C\):

\[
x_C = \arg\min_x L_i(x_R, x) - L_o(f(x_R), f(x))
\]

with \(L_i\) and \(L_o\) some loss that measures the distance between inputs and outputs, respectively, and \(f\) the classifier in question. However, optimizing this objective can be problematic because it contains competing losses and does not guarantee that the generated counterfactual \(x_C\) is part of the data distribution \(p(x)\). Current approaches try to remedy this by incorporating additional regularizers in the objective (Liu et al., 2019; Verma et al., 2020), such as adversarial losses that aim to ensure that the counterfactual \(x_C\) is not distinguishable from a sample \(x \sim p(x)\) (Barredo-Arrieta and Del Ser, 2020; Liu et al., 2019). However, this does not address the core problem of competing objectives and will result in a compromise between obtaining in-distribution samples, maximizing class differences, and minimizing input differences. Interpreting the presented work in this context, we circumvent this issue by dropping the input similarity loss in the generation of counterfactuals and instead enforce similarity post-hoc, similar to the strategy used by Mothilal et al. (2020).

A closely related work addressing counterfactual interpretability is the method presented by Narayanaswamy et al. (2020). Similar to ours, this method uses a cycle-GAN to generate counterfactuals for DNN interpretability. However, this method differs in that the cycle-GAN is applied multiple times to a particular input in order to increase the visual differences in the real and counterfactual images for hypothesis generation. Subsequently, the found features are confirmed by contrasting the original classifiers performance with one that is trained on the discovered features. Similar to other previous methods, this does not lead to attribution maps or an objective evaluation of feature importance.

**Attribution and Counterfactuals** Closest to our approach is the work by Wang and Vasconcelos (2020), which proposes to combine attribution and counterfactual explanations. This work introduces a novel family of so-called discriminative explanations that also leverage attribution on a real and counterfactual image in addition to confidence scores from the classifier to derive attributions for the real and counterfactual image that show highly discriminative features. In contrast to our work, this approach requires calculation of three different attribution maps, which are subsequently combined to produce a discriminative explanation. In addition, this method does not generate new counterfactuals using a generative model, but instead selects a real image from a different class. On one hand this is advantageous because it does not depend on the generator’s performance, but on the other hand this does not allow creating hybrid images for the evaluation of attribution maps.
Another relevant work is presented by Goyal et al. (2019). Similar to our work, the authors devise a method to generate counterfactual visual explanations by searching for a feature pair in two real images of different classes that, if swapped, influences the classification decision. To this end, they propose an optimization procedure that searches for the best features to swap, utilizing the networks feature representations. In contrast to our work, the usage of real (instead of generated) counterfactuals can lead to more artifacts during the replacement of features. In addition, our work supports the generation of attribution maps and features a procedure for the quantitative evaluation of the explanations.

**Attribution Evaluation** Our work differs notably from the current state of the art as it enables quantitative evaluation of the generated attributions by copy-pasting features from a paired image set (the real and the counterfactual). Prior work evaluated the importance of highlighted features by removing them (Samek et al., 2016). However, it has been noted that this strategy is problematic because it is unclear whether any observed performance degradation is due to the removal of relevant features or because the new sample comes from a different distribution. As a result, strategies to remedy this issue have been proposed, for example by retraining classifiers on the modified samples (Hooker et al., 2018). Instead of removing entire features, in this work we replace them with their corresponding counterfactual features.

### 3 Method

The method we propose combines counterfactual interpretability with discriminative attribution methods to find and highlight the most important features between images of two distinct classes \( i \) and \( j \), given a pretrained classifier \( f \). For that, we first generate for a given input image \( x_R \) of class \( i \) a counterfactual image \( x_C \) of class \( j \). We then use a discriminative attribution method to find the attribution map of the classifier for this pair of images. As we will show qualitatively and quantitatively in Section 4, using paired images results in attribution maps of higher quality. Furthermore, the use of a counterfactual image gives rise to an objective evaluation procedure for attribution maps.

In the next sections we describe (1) our choice for generating counterfactual images, (2) the derivation of discriminative attribution methods from existing baseline attribution methods, and (3) how to use counterfactual images to evaluate attribution maps. We denote with \( f \) a pretrained classifier with \( N \) output classes, input images \( x \in \mathbb{R}^{h \times w} \), and output vector \( f(x) = y \in [0,1]^N \) with \( \sum_i y_i = 1 \).

#### 3.1 Creation of Counterfactuals

We train a cycle-GAN (Zhu et al., 2017) for each pair of image classes \( i \neq j \in \{0,1,\ldots,N-1\} \), which enables translation of images of class \( i \) into images of class \( j \) and vice versa. We perform this translation for each image of class \( i \) and each target class \( j \neq i \) to obtain datasets of paired images \( D_{i\rightarrow j} = \{ (x_R^k, x_C^k) | k = 1,\ldots,n(i) \} \), where \( x_R^k \) denotes the \( k \)th real image of class \( i \) and \( x_C^k \) its counterfactual of class \( j \). We then test for each image in the dataset whether the translation was successful by classifying the counterfactual image \( x_C \) and reject a sample pair whenever \( f(x_C) < \theta \), with \( \theta \) a threshold parameter (in the rest of this work we set \( \theta = 0.8 \), except otherwise specified).

This procedure results in a dataset of paired images, where the majority of the differences between an image pair is expected to be relevant for the classifiers decision, i.e., we retain formerly present non-discriminatory distractors such as orientation, lighting, or background. We encourage that the translation makes as little changes as necessary by choosing a Res-Net (He et al., 2016) architecture for the cycle-GAN generator, which is able to trivially learn the identity function.

#### 3.2 Discriminative Attribution from Counterfactuals

The datasets \( D_{i\rightarrow j} \) are already useful to visualize data-intrinsic class differences (see Fig. 5 for examples). However, we wish to understand which input features the classifier \( f \) makes use of. Specifically, we are interested in finding the smallest binary mask \( m \), such that swapping the contents of \( x_C \) with \( x_R \) within this mask changes the classification under \( f \).

To find \( m \), we repurpose existing attribution methods that are amendable to be used with a reference image. The goal of those methods is to produce attribution maps \( a \), which we convert into a binary mask via thresholding. A natural choice for our purposes are so-called baseline attribution methods, which derive attribution maps by contrasting an input image with a baseline sample (e.g., a zero
image). In the following, we review suitable attribution methods and derive discriminative versions that use the counterfactual image as their baseline. We will denote the discriminative versions with the prefix $D$.

### 3.2.1 Input * Gradients

One of the first and simplest attribution methods is *Input * Gradients (INGRADS) (Shrikumar et al., 2016; Simonyan et al., 2014), which is motivated by the first order Taylor expansion of the output class with respect to the input around the zero point:

$$\text{INGRADS}(x) = |\nabla_x f(x)_i \cdot x|,$$

where $i$ is the class for which an attribution map is to be generated. We derive an explicit baseline version for the discriminatory attribution of the real $x_R$ and its counterfactual $x_C$ by choosing $x_C$ as the Taylor expansion point:

$$D\text{-INGRADS}(x_R, x_C) = |\nabla_x f(x)_{i}^{x=x_C} \cdot (x_C - x_R)|,$$

where $j$ is the classes of the counterfactual image.

### 3.2.2 Integrated Gradients

*Integrated Gradients* (IG) is an explicit baseline attribution method, where gradients are accumulated along the straight path from a baseline input $x_0$ to the input image $x$ to generate the attribution map (Sundararajan et al., 2017). Integrated gradients along the $k$th dimension are given by:

$$\text{IG}_k(x) = (x - x_0)_k \cdot \int_{\alpha=0}^{1} \frac{\partial f(x_0 + \alpha(x - x_0))_i}{\partial x_k} d\alpha.$$

We derive a discriminatory version of IG by replacing the baseline as follows:

$$D\text{-IG}_k(x_R, x_C) = (x_C - x_R)_k \cdot \int_{\alpha=0}^{1} \frac{\partial f(x_R + \alpha(x_C - x_R))_i}{\partial x_k} d\alpha.$$

### 3.2.3 Deep Lift

*Deep Lift* (DL) is also an explicit baseline attribution method which aims to compare individual neurons activations of an input w.r.t. a reference baseline input (Shrikumar et al., 2016). It can be expressed in terms of the gradient in a similar functional form to IG:

$$\text{DL}(x) = (x - x_0) \cdot F_{DL},$$

where $F_{DL}$ is some function of the gradient of the output (see Ancona et al. (2018) for the full expression). The discriminative attribution we consider is simply:

$$D\text{-DL}(x_R, x_C) = (x_C - x_R) \cdot F_{DL}.$$

### 3.2.4 GradCAM

*GradCAM* (GC) is an attribution method that considers the gradient weighted activations of a particular layer, usually the last convolutional layer, and propagates this value back to the input image (Selvaraju et al., 2017). We denote the activation of a pixel $(u, v)$ in layer $l$ with size $(h, w)$ and channel $k$ by $C_{k,u,v}^l$, and write the gradient w.r.t. the output $y$ as:

$$\nabla_{C_{k,u,v}^l} y = \left(\frac{dy}{dC_{k,0,0}^l}, \frac{dy}{dC_{k,1,0}^l}, \frac{dy}{dC_{k,2,0}^l}, \ldots, \frac{dy}{dC_{k,h,w}^l}\right).$$

The original GC is then defined as:

$$\text{GC}(x) = \text{ReLU}\left(\sum_k \nabla_{C_{k,u,v}^l} y \cdot C_{k,u,v}^l\right) = \text{ReLU}\left(\sum_k \sum_{u,v} \frac{dy}{dC_{k,u,v}^l} C_{k,u,v}^l\right) = \text{ReLU}\left(\sum_k \alpha_k C_k\right).$$
where we omitted the layer index $l$ for brevity. Each term $\frac{\partial y}{\partial C_{k,u,v}}$ is the contribution of pixel $u,v$ in channel $k$ to the output classification score $y$ under a linear model. GC utilizes this fact and projects the layer attribution from layer $l$ back to the input image, generating the final attribution map.

In contrast to the setting considered by GC, we have access to a matching pair of real and counterfactual images $x_R$ and $x_C$. We extend GC to consider both feature maps $C^{x_R}$ and $C^{x_C}$ by treating GC as an implicit zero baseline method similar to INGRADS:

$$D\text{-}GC_k(x_R, x_C) = \frac{\partial y}{\partial C_{k}} C_{k}C^{x_C} (C_k(x_C) - C_k(x_R)) .$$

Averaging those gradients over feature maps $k$, and projecting the activations back to image space then highlights pixels that are most discriminative for a particular pair:

$$D\text{-}GC_p(x_R, x_C) = \mathcal{P} \sum_k D\text{-}GC_k(x_R, x_C) ,$$

where $\mathcal{P}$ is the projection matrix from feature space $C$ to input space $X$. Note that in contrast to GC, we use the absolute value of the output attribution, as we do not apply ReLU activations to layer attributions.

Because feature maps can be of lower resolution than the input space, GC tends to produce coarse attribution maps (Selvaraju et al., 2017). To address this issue it is often combined with Guided Backpropagation (GBP), a method that uses the gradients of the output class w.r.t. the input image as the attribution map (Springenberg et al., 2014). During the backwards pass, all values $< 0$ at each ReLU non-linearity are then discarded to only retain positive attributions.

Guided GradCAM (GGC) uses this strategy to sharpen the attribution of GC via element-wise multiplication of the attribution maps (Selvaraju et al., 2017). For the baseline versions we thus consider multiplication of $D\text{-}GC$ with the GBP attribution maps:

$$GBP(x) = \nabla_C f(x) , \text{ with } \nabla \text{ReLU} > 0$$

$$GGC(x) = GC(x) \cdot GBP(x)$$

$$D\text{-}GGC(x_R, x_C) = D\text{-}GC(x_R, x_C) \cdot GBP(x_R).$$

### 3.3 Evaluation of Attribution Maps

The discriminative attribution map $a$ obtained for pair of images $(x_R, x_C)$ can be used to quantify the causal effect of the attribution. Specifically, we can copy the area highlighted by $a$ from the real image $x_R$ of class $i$ to the counterfactual image $x_C$ of class $j$, resulting in a hybrid image $x_H$. If the attribution accurately captures class-relevant features, we would expect that the classifier $f$ assigns a high probability to $x_H$ being of class $i$.

![Figure 2: Evaluation procedure for discriminative attribution methods: Given the real image $x_R$ of class $i$ and its counterfactual $x_C$ of class $j$, we generate a sequence of binary masks $m$ by applying different thresholds to the attribution map $a$. Those masks are then used to generate a sequence of hybrid images $x_H$. The plot shows the change in classifier prediction $\Delta f(x_H)_i = f(x_H)_i - f(x_C)_i$ over the size of the mask $m$ (normalized between 0 and 1). The DAC score is the area under the curve, i.e., a value between 0 and 1. Higher DAC scores are better and indicate that a discriminative attribution method found small regions that lead to the starkest change in classification.](image-url)
Figure 3: Example images of datasets SYNAPSES and Disc. SYNAPSES consists of electron microscopy images of synapses. Each class is defined by the neurotransmitter the synapse releases. Disc is a synthetic dataset we designed in order to highlight failure cases of popular attribution methods. We consider two subsets: Disc-A shows triangles in each image and classes are defined by the parity of the number of triangles. Disc-B consists of images showing triangles, squares, and disks. Each class is one combination of two shapes.

The ability to create those hybrid images is akin to an intervention, and has two important practical implications: First, it allows us to find a minimal binary mask that captures the most class-relevant areas for a given input image. Second, we can compare the change in classification score for hybrids derived from different attribution maps. This allows us to compare different methods in an objective manner, following the intuition that an attribution map is better, if it changes the classification with less pixels changed.

To find a minimal binary mask $m_{\text{min}}$, we search for a threshold of the attribution map $a$, such that the mask score $\Delta f(x_H) = f(x_H) - f(x_C)$ (i.e., the change in classification score) is maximized while the size of the mask is minimized, i.e., $m_{\text{min}} = \arg \min_m |m| - \Delta f(x_H)$ (where we omitted the dependency of $x_H$ on $m$ for brevity). In order to minimize artifacts in the copying process we also apply a morphological closing operation with a window size of 10 pixels followed by a Gaussian Blur with $\sigma = 11$ px. The final masks highlight the relevant class discriminators by showing the user the counterfactual features, the original features they are replaced with, and the corresponding mask score $\Delta f(x_H)$, indicating the quantitative effect of the replacement on the classifier. See Fig. 5 for example pairs and corresponding areas $m_{\text{min}}$.

Furthermore, by applying a sequence of thresholds for the attribution map $a$, we derive an objective evaluation procedure for a given attribution map: For each hybrid image $x_H$ in the sequence of thresholds, we consider the change in classifier prediction relative to the size of the mask that has been used to create the hybrid. We accumulate the change in classifier prediction over all mask sizes to derive our proposed DAC score. This procedure is explained in detail in Fig. 2 for a single pair of images. When reporting the DAC score for a particular attribution method, we average the single DAC scores over all images, and all distinct pairs of classes.

4 Experiments

We evaluate the presented method on four datasets: MNIST (LeCun and Cortes, 2010), SYNAPSES (Eckstein et al., 2020)¹ and two versions of a synthetic dataset that we call Disc-A and Disc-B (see Fig. 3 for an overview).

SYNAPSES A real world biological dataset, consisting of 128 × 128px electron microscopy images of synaptic sites in the brain of Drosophila melanogaster. Each image is labelled with a functional property of the synapse, namely the neurotransmitter it releases (the label was acquired using immunohistochemistry labelling, see Eckstein et al. (2020) for details). This dataset is of particular interest for interpretability, since a DNN can recover the neurotransmitter label from the images with high accuracy, but human experts are not able to do so. Interpretability methods like the one

¹Dataset kindly provided by the authors of Eckstein et al. (2020).
Figure 4: Quantitative evaluation of discriminative ($D$ - solid) and corresponding original ($S$ for “single input” - dashed) attribution methods over four datasets. Corresponding $D$ and $S$ versions of the same method are shown in the same color. For each, we plot the average change of classifier prediction $\Delta f(x_H)^{D} = f(x_H) - f(x_C)$ as a function of mask size $m \in [0, 1]$. In addition we show performance of the two considered baselines: masks derived from random attribution maps (random - red, dotted) and mask derived from the residual of the real and counterfactual image (residual - black, dotted). On all considered datasets all versions of $D$ attribution outperform their $S$ counterparts. All experiments are performed with VGG architectures. For ResNet results of Mnist and Disc see supplement.

presented here can thus be used to gain insights into the relation between structure (from the electron microscopy image) and function (the neurotransmitter released).

Disc-A and Disc-B  Two synthetic datasets with different discriminatory features of different difficulty. Each image is $128 \times 128$ px in size and contains spheres, triangles or squares. For Disc-A, the goal is to correctly classify images containing an even or odd number of triangles. Disc-B contains images that show exactly two of the three available shapes and the goal is to predict which shape is missing (e.g., an image with only triangles and squares is to be classified as “does not contain spheres”). This dataset was deliberately designed to investigate attribution methods in a setting where the discrimination depends on the absence of a feature.

Training  For Mnist and Disc, we train a VGG and ResNet for 100 epochs and select the epoch with highest accuracy on a held out validation dataset. For Synapses we adapt the 3D-VGG architecture from Eckstein et al. (2020) to 2D and train for 500,000 iterations. We select the iteration with the highest validation accuracy for testing. For each dataset we train one cycle-GAN for 200 epochs, on each class pair and on the same training set the respective classifier was trained on (the full network specifications are given in the supplement).

Results  Quantitative results (in terms of the DAC score, see Section 3.3) for each investigated attribution method are shown in Fig. 4 and Table 1. In summary, we find that attribution maps generated from the proposed discriminative attribution methods consistently outperform their original versions in terms of the DAC score. This observation also holds visually: the generated masks from discriminative attribution methods are smaller and more often highlight the main discriminatory parts of a considered image pair (see Fig. 5). In particular, the proposed method substantially outperforms the considered random baseline, whereas standard attribution methods sometimes fail to do so (e.g., GC on dataset Synapses). Furthermore, on Mnist and Disc-A, the mask derived from the residual of real and counterfactual image is already competitive with the best considered methods and outperforms standard attribution substantially. However, for more complex datasets such as Synapses the residual becomes less accurate in highlighting discriminative features. Here, the discriminatory attributions outperform all other considered methods.

5 Discussion

This work demonstrates that the combination of counterfactual interpretability with suitable attribution methods is more accurate in extracting key discriminative features between class pairs than standard methods. While the method succeeds in the presented experiments, it comes with a number of limitations. It requires the training of cycle-GANs, one for each pair of output classes. Thus training time and compute cost scale quadratically in the number of output classes and it is therefore not
Table 1: Summary of DAC scores for each investigated method on the three datasets MNIST, SYNAPSES, and Disc (two versions) corresponding to 4. Best results are highlighted. For extended results with ResNet architectures see supplement.

Figure 5: Samples from the best performing method pair on SYNAPSES and Disc-A and B. Discriminative attribution methods are able to highlight the key discriminative features while vanilla versions often fail to do so (e.g., a subtle intensity change in the synaptic cleft in the top rows). Further qualitative results (including the other considered datasets) can be found in the supplement.
feasible for classification problems with a large number of classes. Furthermore, the translation from the real to the counterfactual image could fail for a large fraction of input images, i.e., $f(x_C) \neq j$. In this work, we only consider those image pairs where translation was successful, as we focus on extracting knowledge about class differences from the classifier. For applications that require an attribution for each input image this approach is not suitable. An additional concern is that focusing only on images that have a successful translation may bias the dataset we consider and with it the results. GANs are known to exhibit so called mode collapse (Che et al., 2016; Salimans et al., 2016), meaning they focus on only a small set of modes of the full distribution. As a consequence, the method described here may miss discriminatory features present in other modes. Using a cycle-GAN is not possible in all image domains. Image classes need to be sufficiently similar in appearance for the cycle-GAN to work, and translating, e.g., an image of a mouse into an image of a tree is unlikely to work and produce meaningful attributions. However, we believe that the generation of masks in combination with the corresponding mask score is superior for interpreting DNN decision boundaries than classical attribution maps and suggest the usage of cycle-GAN baselines for attribution in cases where a fine grained understanding of class differences is sought.

Although we present this work in the context of understanding DNNs and the features they make use of, an uncritical adaptation of this and other similar interpretability methods can potentially lead to ethical concerns. As an example, results should be critically evaluated when using this method to interpret classifiers that have been trained to predict human behaviour, or demographic and socioeconomic features. As with any data-heavy method, it is important to realize that results will be reflective of data- and model-intrinsic biases. As such, an interpretability method like the one we present here can at most identify a correlation between input features and labels, but not true causal links. The method presented here should therefore not be used to “proof” that a particular feature leads to a particular outcome. Such claims should be met with criticism to prevent agenda-driven narratives of malicious actors.

References
Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., and Kim, B. (2018). Sanity checks for saliency maps. arXiv preprint arXiv:1810.03292.
Alvarez-Melis, D. and Jaakkola, T. S. (2018). On the robustness of interpretability methods. arXiv preprint arXiv:1806.08049.
Ancona, M., Ceolini, E., Öztireli, C., and Gross, M. (2018). Towards better understanding of gradient-based attribution methods for deep neural networks. In International Conference on Learning Representations.
Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PloS one, 10(7):e0130140.
Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., and Müller, K.-R. (2010). How to explain individual classification decisions. The Journal of Machine Learning Research, 11:1803–1831.
Barredo-Arrieta, A. and Del Ser, J. (2020). Plausible counterfactuals: Auditing deep learning classifiers with realistic adversarial examples. In 2020 International Joint Conference on Neural Networks (IJCNN), pages 1–7. IEEE.
Che, T., Li, Y., Jacob, A. P., Bengio, Y., and Li, W. (2016). Mode regularized generative adversarial networks. arXiv preprint arXiv:1612.02136.
Dabkowski, P. and Gal, Y. (2017). Real time image saliency for black box classifiers. arXiv preprint arXiv:1705.07857.
Eckstein, N., Bates, A. S., Du, M., Hartenstein, V., Jefferis, G. S., and Funke, J. (2020). Neurotransmitter classification from electron microscopy images at synaptic sites in drosophila. BioRxiv.
Fong, R. C. and Vedaldi, A. (2017). Interpretable explanations of black boxes by meaningful perturbation. In Proceedings of the IEEE International Conference on Computer Vision, pages 3429–3437.
Ghorbani, A., Abid, A., and Zou, J. (2019). Interpretation of neural networks is fragile. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pages 3681–3688.
Goyal, Y., Wu, Z., Ernst, J., Batra, D., Parikh, D., and Lee, S. (2019). Counterfactual visual explanations. In International Conference on Machine Learning, pages 2376–2384. PMLR.
He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778.

Hooker, S., Erhan, D., Kindermans, P.-J., and Kim, B. (2018). A benchmark for interpretability methods in deep neural networks. arXiv preprint arXiv:1806.10758.

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1125–1134.

Kindermans, P.-J., Hooker, S., Adebayo, J., Alber, M., Schütt, K. T., Dähne, S., Erhan, D., and Kim, B. (2019). The (un)reliability of saliency methods. In Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, pages 267–280. Springer.

Kindermans, P.-J., Schütt, K. T., Alber, M., Müller, K.-R., Erhan, D., Kim, B., and Dähne, S. (2017). Learning how to explain neural networks: Patternet and patternattrition. arXiv preprint arXiv:1705.05598.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

LeCun, Y. and Cortes, C. (2010). MNIST handwritten digit database.

Liu, S., Kailkhura, B., Loveland, D., and Han, Y. (2019). Generative counterfactual introspection for explainable deep learning. arXiv preprint arXiv:1907.03077.

Lundberg, S. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. arXiv preprint arXiv:1705.07874.

Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z., and Paul Smolley, S. (2017). Least squares generative adversarial networks. In Proceedings of the IEEE international conference on computer vision, pages 2794–2802.

Martens, D. and Provost, F. (2014). Explaining data-driven document classifications. Mis Quarterly, 38(1):73–100.

Montavon, G., Lapuschkin, S., Binder, A., Samek, W., and Müller, K.-R. (2017). Explaining nonlinear classification decisions with deep taylor decomposition. Pattern Recognition, 65:211–222.

Mothilal, R. K., Sharma, A., and Tan, C. (2020). Explaining machine learning classifiers through diverse counterfactual explanations. In Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, pages 607–617.

Narayanaswamy, A., Venugopalan, S., Webster, D. R., Peng, L., Corrado, G. S., Ruamviboonsuk, P., Bavishi, P., Brenner, M., Nelson, P. C., and Varadarajan, A. V. (2020). Scientific discovery by generating counterfactuals using image translation. In International Conference on Medical Image Computing and Computer-Assisted Intervention, pages 273–283. Springer.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). Model-agnostic interpretability of machine learning. arXiv preprint arXiv:1606.05386.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. arXiv preprint arXiv:1606.03498.

Samek, W., Binder, A., Montavon, G., Lapuschkin, S., and Müller, K.-R. (2016). Evaluating the visualization of what a deep neural network has learned. IEEE transactions on neural networks and learning systems, 28(11):2660–2673.

Samek, W., Montavon, G., Lapuschkin, S., Anders, C. J., and Müller, K.-R. (2021). Explaining deep neural networks and beyond: A review of methods and applications. Proceedings of the IEEE, 109(3):247–278.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE international conference on computer vision, pages 618–626.

Shrikumar, A., Greenside, P., and Kundaje, A. (2017). Learning important features through propagating activation differences. In International Conference on Machine Learning, pages 3145–3153. PMLR.

Shrikumar, A., Greenside, P., Scherhribina, A., and Kundaje, A. (2016). Not just a black box: Learning important features through propagating activation differences. arXiv preprint arXiv:1605.01713.

Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Deep inside convolutional networks: Visualising image classification models and saliency maps.
Smilkov, D., Thorat, N., Kim, B., Viégas, F., and Wattenberg, M. (2017). Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.

Sundararajan, M., Taly, A., and Yan, Q. (2017). Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3319–3328. JMLR. org.

Verma, S., Dickerson, J., and Hines, K. (2020). Counterfactual explanations for machine learning: A review. *arXiv preprint arXiv:2010.10596*.

Wachter, S., Mittelstadt, B., and Russell, C. (2017). Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841.

Wang, P. and Vasconcelos, N. (2020). Scout: Self-aware discriminant counterfactual explanations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8981–8990.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2016). Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232.

Zintgraf, L. M., Cohen, T. S., Adel, T., and Welling, M. (2017). Visualizing deep neural network decisions: Prediction difference analysis. *arXiv preprint arXiv:1702.04595*.
We use the Adam optimizer (Kingma and Ba, 2014) with momentum squares loss (LSGAN (Mao et al., 2017)), a batch size of one, instance normalization and normal initialization. We extend the cycle-GAN implementation from https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

| Operation       | Tensor Size |
|-----------------|-------------|
| input image     | (128, 128)  |
| Conv2d, size (3, 3) | (12, 128, 128) |
| BatchNorm2d     | (12, 128, 128) |
| ReLU            | (12, 128, 128) |
| Conv2d, size (3, 3) | (12, 128, 128) |
| BatchNorm2d     | (12, 128, 128) |
| ReLU            | (12, 128, 128) |
| MaxPool2d, size (2, 2) | (12, 64, 64) |
| Conv2d, size (3, 3) | (24, 64, 64) |
| BatchNorm2d     | (24, 64, 64) |
| ReLU            | (24, 64, 64) |
| Conv2d, size (3, 3) | (24, 64, 64) |
| BatchNorm2d     | (24, 64, 64) |
| ReLU            | (24, 64, 64) |
| MaxPool2d, size (2, 2) | (24, 32, 32) |
| Conv2d, size (3, 3) | (48, 32, 32) |
| BatchNorm2d     | (48, 32, 32) |
| ReLU            | (48, 32, 32) |
| Conv2d, size (3, 3) | (48, 32, 32) |
| BatchNorm2d     | (48, 32, 32) |
| ReLU            | (48, 32, 32) |
| MaxPool2d, size (2, 2) | (48, 16, 16) |
| Conv2d, size (3, 3) | (96, 16, 16) |
| BatchNorm2d     | (96, 16, 16) |
| ReLU            | (96, 16, 16) |
| Conv2d, size (3, 3) | (96, 16, 16) |
| BatchNorm2d     | (96, 16, 16) |
| ReLU            | (96, 16, 16) |
| MaxPool2d, size (2, 2) | (96, 8, 8) |
| Linear          | (4096)      |
| ReLU            | (4096)      |
| Dropout         | (4096)      |
| Linear          | (4096)      |
| ReLU            | (4096)      |
| Dropout         | (4096)      |
| Linear          | (4096)      |
| Linear          | (k)         |

(a) VGG architecture used for the SYNAPSES (k = 6), Disc-A (k = 2), and Disc-B (k = 3) dataset.

| Operation       | Tensor Size |
|-----------------|-------------|
| input image     | (28, 28)    |
| Conv2d, size (3, 3) | (12, 28, 28) |
| BatchNorm2d     | (12, 28, 28) |
| ReLU            | (12, 28, 28) |
| Conv2d, size (3, 3) | (12, 28, 28) |
| BatchNorm2d     | (12, 28, 28) |
| ReLU            | (12, 28, 28) |
| MaxPool2d, size (2, 2) | (12, 14, 14) |
| Conv2d, size (3, 3) | (24, 14, 14) |
| BatchNorm2d     | (24, 14, 14) |
| ReLU            | (24, 14, 14) |
| Conv2d, size (3, 3) | (24, 14, 14) |
| BatchNorm2d     | (24, 14, 14) |
| ReLU            | (24, 14, 14) |
| MaxPool2d, size (2, 2) | (24, 7, 7) |
| Conv2d, size (3, 3) | (48, 7, 7) |
| BatchNorm2d     | (48, 7, 7) |
| ReLU            | (48, 7, 7) |
| Conv2d, size (3, 3) | (48, 7, 7) |
| BatchNorm2d     | (48, 7, 7) |
| ReLU            | (48, 7, 7) |
| Conv2d, size (3, 3) | (96, 7, 7) |
| BatchNorm2d     | (96, 7, 7) |
| ReLU            | (96, 7, 7) |
| Conv2d, size (3, 3) | (96, 7, 7) |
| BatchNorm2d     | (96, 7, 7) |
| ReLU            | (96, 7, 7) |
| Linear          | (4096)      |
| ReLU            | (4096)      |
| Dropout         | (4096)      |
| Linear          | (4096)      |
| ReLU            | (4096)      |
| Dropout         | (4096)      |
| Linear          | (4096)      |
| Linear          | (10)        |

(b) VGG architecture used for the MNIST dataset.

Table 2: VGG classifier network architectures.

A Training Details

A.1 Network Architectures

**Cycle-GAN** We extend the cycle-GAN implementation from https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix for our purposes. For all experiments we use a 9-block ResNet generator and a 70 × 70 PatchGAN (Isola et al., 2017) discriminator. For training we use a least squares loss (LSGAN (Mao et al., 2017)), a batch size of one, instance normalization and normal initialization. We use the Adam optimizer (Kingma and Ba, 2014) with momentum $\beta_1 = 0.5$ and a learning rate of 0.0002 with a linear decay to zero after the first 100 epochs.
The most significant part of the compute costs come from training the cycle-GANs. For each
results: All discriminative attribution methods outperform their counterparts in terms of DAC-score.

For the training of the VGG network on the Synapse dataset, we use the same strategy (including augmentations) as described in Eckstein et al. (2020), with the only difference being that we consider 2D images instead of 3D volumes. We did not attempt to train a ResNet on the Synapse dataset.

For the training of the VGG and ResNet architectures on the MNIST and DISC datasets we did not make use of augmentations and trained each network for 100 epochs with a batch size of 32 using the Adam optimizer (learning rate $10^{-4}$).

### A.2 Compute

The most significant part of the compute costs come from training the cycle-GANs. For each experiment, cycle-GAN training for 200 epochs took around 5 days on a single RTX 2080Ti GPU. For MNIST experiments we trained a total of 45 cycle GANs, 15 for Synapse, and 4 for DISC. In total this results in roughly 320 GPU-days for cycle-GAN training. In contrast, attribution and mask generation is comparatively cheap and takes between 1-3 hours on 20 RTX 2080Ti GPUs for each dataset, resulting in 60 GPU hours for each experiment and 15 GPU days in total.

### B Extended Results for ResNet Architectures

In addition to the results using VGG architectures in the main text, below we show additional results for ResNet architectures on MNIST and DISC-B (see Fig. 6 and Table 4). We do not show results for DISC-A, because all considered ResNet architectures failed to achieve more than chance level accuracy on the validation dataset. Since our goal is to understand what the classifier learned about class differences, using a network that did not successfully learn to classify will not produce meaningful results.

The shown results for ResNet architectures follow the same pattern as observed in the main VGG results: All discriminative attribution methods outperform their counterparts in terms of DAC-score.

| Operation       | Tensor Size   | Operation       | Tensor Size   |
|-----------------|--------------|-----------------|--------------|
| input image     | (128, 128)   | input image     | (28, 28)     |
| Conv2d, size (3, 3) | (12, 128, 128) | Conv2d, size (3, 3) | (12, 28, 28) |
| BatchNorm2d     | (12, 128, 128) | BatchNorm2d     | (12, 28, 28) |
| ReLU            | (12, 128, 128) | ReLU            | (12, 28, 28) |
| ResBlock, stride (2, 2) | (12, 64, 64)   | ResBlock, stride (2, 2) | (12, 14, 14) |
| ResBlock        | (12, 64, 64)   | ResBlock        | (12, 14, 14) |
| ResBlock, stride (2, 2) | (24, 32, 32)   | ResBlock, stride (2, 2) | (24, 7, 7)   |
| ResBlock        | (24, 32, 32)   | ResBlock        | (24, 7, 7)   |
| ResBlock, stride (2, 2) | (48, 16, 16)   | ResBlock, stride (2, 2) | (48, 3, 3)   |
| ResBlock        | (48, 16, 16)   | ResBlock        | (48, 3, 3)   |
| ResBlock, stride (2, 2) | (96, 8, 8)     | ResBlock, stride (2, 2) | (96, 1, 1)   |
| ResBlock        | (96, 8, 8)     | ResBlock        | (96, 1, 1)   |
| Linear          | (4096)        | Linear          | (4096)       |
| ReLU            | (4096)        | ReLU            | (4096)       |
| Dropout         | (4096)        | Dropout         | (4096)       |
| Linear          | (4096)        | Linear          | (4096)       |
| ReLU            | (4096)        | ReLU            | (4096)       |
| Dropout         | (4096)        | Dropout         | (4096)       |
| Linear          | (k)           | Linear          | (10)         |

(a) ResNet architecture used for the DISC-A ($k = 2$) and DISC-B ($k = 3$) dataset.

Table 3: ResNet classifier network architectures.
Figure 6: Quantitative evaluation of discriminative (D - solid) and corresponding original (S for "single input" - dashed) attribution methods for MNIST and Disc-B using a ResNet architecture. Corresponding D and S versions of the same method are shown in the same color. For each, we plot the average change of classifier prediction $\Delta f(x_H)_i^k = f(x_H)_i - f(x_C)_i$ as a function of mask size $m \in [0, 1]$. In addition we show performance of the two considered baselines: masks derived from random attribution maps (random - red, dotted) and mask derived from the residual of the real and counterfactual image (residual - black, dotted). On all considered datasets all versions of D attribution outperform their S counterparts.

For MNIST, the overall best performing method is the residual, which already performed well for VGG experiments. This is a consequence of the sparsity and simplicity of MNIST and can be observed less drastically for Disc as well. The changes the cycle-GAN introduces are often minimal, and thus the residual is already an accurate attribution. However, in general, the residual is not a good choice for an attribution map as intensity differences between classes do not generally correlate with feature importance. This is particularly noticeable in the experiments on the more challenging Synapset experiments (see main text).

### Disc Dataset

The Disc dataset was specifically designed to highlight the advantage of discriminative attribution over vanilla attribution. In particular, the discriminatory feature of Disc-A is the parity of the number of triangles in the image. This feature is non-local and it is unclear what vanilla attribution is supposed to highlight. In Disc-B the classes are defined by the absence of a feature, another situation where vanilla attribution is not designed to give a sensible answer and will often highlight all objects in the image, providing little information to the user.

#### Disc-A

For each image we randomly draw an even (class 0) or odd (class 1) number between one and six, indicating the number of triangles to generate. Each triangle has a random size between 20 and 40% of the image size of 128 pixels and a random position. In addition we draw a random intensity value between 120 and 200, a random rotation angle, and additive noise strength before applying Gaussian smoothing to generate different textures. We reject a sample if the fraction of foreground pixels and the total expected area of all shapes (assuming no overlap) is below 90%, thus avoiding strongly overlapping configurations.

#### Disc-B

Similar to Disc-A, we draw a random position, intensity value, rotation and additive noise strength to generate images showing pairs of a triangle and a square, a disk and a square or a disk and
Figure 7: Qualitative samples from the Synapses dataset for all considered methods. \(x_R\) shows a synapse from class Serotonin, \(x_C\) shows a synapse from class Acetylcholine.
Figure 8: Qualitative samples from the SYNAPSES dataset for all considered methods. $x_R$ shows a synapse from class Acetylcholine, $x_C$ shows a synapse from class Octopamine.
Figure 9: Qualitative samples from the Synapses dataset for all considered methods. $x_R$ shows a synapse from class Serotonin, $x_C$ shows a synapse from class Glutamate.
Figure 10: Qualitative samples from the SYNAPSES dataset for all considered methods. $x_R$ shows a synapse from class Dopamine, $x_C$ shows a synapse from class Serotonin.
Figure 11: Qualitative sample from the Mnist dataset for all considered methods.
| Method       | $a$ | $m$ | $m \cdot x_R$ | $m \cdot x_C$ | $x_H$ |
|--------------|-----|-----|---------------|---------------|-------|
| D-IG         | ![Image](image1.png) | ![Image](image2.png) | ![Image](image3.png) | ![Image](image4.png) | ![Image](image5.png) |
| IG           | ![Image](image6.png) | ![Image](image7.png) | ![Image](image8.png) | ![Image](image9.png) | ![Image](image10.png) |
| D-DL         | ![Image](image11.png) | ![Image](image12.png) | ![Image](image13.png) | ![Image](image14.png) | ![Image](image15.png) |
| DL           | ![Image](image16.png) | ![Image](image17.png) | ![Image](image18.png) | ![Image](image19.png) | ![Image](image20.png) |
| D-INGR.      | ![Image](image21.png) | ![Image](image22.png) | ![Image](image23.png) | ![Image](image24.png) | ![Image](image25.png) |
| INGR.        | ![Image](image26.png) | ![Image](image27.png) | ![Image](image28.png) | ![Image](image29.png) | ![Image](image30.png) |
| D-GC         | ![Image](image31.png) | ![Image](image32.png) | ![Image](image33.png) | ![Image](image34.png) | ![Image](image35.png) |
| GC           | ![Image](image36.png) | ![Image](image37.png) | ![Image](image38.png) | ![Image](image39.png) | ![Image](image40.png) |
| D-GGC        | ![Image](image41.png) | ![Image](image42.png) | ![Image](image43.png) | ![Image](image44.png) | ![Image](image45.png) |
| GGC          | ![Image](image46.png) | ![Image](image47.png) | ![Image](image48.png) | ![Image](image49.png) | ![Image](image50.png) |
| RES.         | ![Image](image51.png) | ![Image](image52.png) | ![Image](image53.png) | ![Image](image54.png) | ![Image](image55.png) |
| RND.         | ![Image](image56.png) | ![Image](image57.png) | ![Image](image58.png) | ![Image](image59.png) | ![Image](image60.png) |

Figure 12: Qualitative sample from the Disc-A dataset for all considered methods.
Figure 13: Qualitative sample from the Disc-B dataset for all considered methods.
a triangle. We reject a sample if the fraction of foreground pixels and the total expected area of all shapes (assuming no overlap) is below 90%.

D Code and Data Availability

All code, datasets, checkpoints, and instructions needed to reproduce the presented results are available at https://dac-method.github.io.