Cycle-Consistent Counterfactuals by Latent Transformations

Saeed Khorram, Li Fuxin
Collaborative Robotics and Intelligent Systems (CoRIS) Institute
Oregon State University
{khorrams, lif}@oregonstate.edu

Abstract

CounterFactual (CF) visual explanations try to find images similar to the query image that change the decision of a vision system to a specified outcome. Existing methods either require inference-time optimization or joint training with a generative adversarial model which makes them time-consuming and difficult to use in practice. We propose a novel approach, Cycle-Consistent Counterfactuals by Latent Transformations (C3LT), which learns a latent transformation that automatically generates visual CFs by steering in the latent space of generative models. Our method uses cycle consistency between the query and CF latent representations which helps our training to find better solutions. C3LT can be easily plugged into any state-of-the-art pretrained generative network. This enables our method to generate high-quality and interpretable CF images at high resolution such as those in ImageNet. In addition to several established metrics for evaluating CF explanations, we introduce a novel metric tailored to assess the quality of the generated CF examples and validate the effectiveness of our method on an extensive set of experiments.

1. Introduction

With convolutional neural networks (CNNs) revolutionizing the space of automatic visual recognition, there have been many approaches attempting to better explain the inner workings of CNNs, including attribution maps [21, 41], concept-based explanations [9, 36], rule-based explanations [8], prototype-based explanations [5], etc. However, when presented to humans, those kinds of explanations were not necessarily easy to process. Recently, a substantial user study [16] shows that Grad-CAM [41], LIME superpixels [36], etc., were not as informative to humans as simple nearest neighbors from the training set.

Those findings suggest that humans prefer to see examples that are just similar to the natural images rather than heatmaps, superpixels, etc. and *counterfactual (CF) explanations* [7, 11, 28, 30, 45, 46] might be more useful in helping humans to understand deep networks. CF explanations show humans examples that are similar to the explanation subject but deep networks predict them as a different category. Such explanations have also been advocated by social scientists [27, 45] as a preferred mode of explanation.

This paper mainly deals with CF explanations in the visual domain. CF explanation in the visual domain is more difficult to generate than categorical inputs where one can simply search for adversarial examples [28, 30, 45]. Methods that directly optimize for perturbations in the input space [7] often lead to adversarial solutions [43], which manipulate CNN predictions with imperceptible changes. Adversarial examples are usually off the data manifold, where CNNs are fooled because they do not generalize to the kinds of data that have never been seen in training. In addition, finding and replacing patches of features from images in the CF class to a query one [11] also moves the images off the natural image manifold by creating irregular edges.

Successful CF explanations usually avoid being adversarial by staying on the same data manifold the network has been trained on. Hence, prior work usually utilizes a generative model such as a generative adversarial network (GAN) or variational autoencoder (VAE) that ensures the generated CF example lies on the data manifold. For example, ExplainGAN [39] jointly trains a GAN for each category along with a mask generator that generates a masked region from the latent code of the image, so that after the masked region is transformed the image would be classified as another category by the CNN. Some other algorithms [37] optimize for latent codes of a VAE model that will generate an image similar to the original one yet classified as another category.

Despite these prior work, it remains difficult to apply CF explanations in practice. One can consider two realistic use cases for explanation algorithms. The first is *debugging*, where users attempt to check why CNN is making a certain wrong classification. The second is *knowledge gathering*, where users may try to utilize explanations to understand subtle differences between two classes. In both cases, it would be beneficial for the user to quickly churn through many examples to help building their mental model. Even better, the user may want to make some realistic edits (e.g. based on GANs)) to the image and then obtain a new CF out of the edited image. In those cases, it would be ideal if CF images can be generated *on-the-fly*. However, most previous approaches solve an optimization for each image [7,11,23,26,30,38], which often makes generating CF examples *time-consuming*.

In this paper, we propose a novel approach that optimizes for a nonlinear transformation in the latent space. The transformation morphs the latent code of an input image into a CF latent vector that can be decoded into an image which looks similar to the original one, but has semantically meaningful, perceptible differences so that CNN classifies it as another category. Different from [39], our approach does not require joint training with GANs. It utilizes a pretrained generative model (GAN/VAE) hence can easily adapt to current and future generative algorithms that are being proposed every day due to significant ongoing research. As an example, this enables our framework to go beyond simple datasets and generate high-resolution images, e.g. ImageNet, with the current GAN algorithms available now. We further adopt a cycle-consistency loss function [50] that improves the consistency and performance of our approach.

Furthermore, we evaluate our approach comprehensively in a quantitative manner. For CF explanations, literature has suggested certain properties to be desirable [29,44]:

I VALIDITY. The model should assign the CF examples $\boldsymbol{x}'$ to the CF class c' in order to be valid.

II PROXIMITY. The CF examples $\boldsymbol{x}'$ should stay as close as possible (in terms of some distance function) to the original query instance $\boldsymbol{x}$.

III SPARSITY. Minimal number of query features should be perturbed in order to generate CF examples.

IV REALISM. The CF examples should lie close to the data manifold so that it appears realistic.

V SPEED. The CF explanations should be generated in interactive speed in order to be deployed in real-world applications.

For example, adversarial examples may be valid CFs, but they fail in terms of realism. We propose to use a set of metrics that comprehensively measure all these aspects, including a novel metric that inspects the quality of the CF examples across a series of changes.

Below we list our contributions in this paper:

- We introduce a novel framework to generate realistic CFs at high resolution by learning a transformation in the latent space of a pretrained generative model.
- We propose a set of novel quantitative evaluation metrics tailored for counterfactual explanations.
- Extensive qualitative and quantitative evaluations show the effectiveness of our method and its capability to generate high-resolution CF images by plugging into existing generative algorithms.

2. Related Work

Counterfactual Visual Explanation. While many of the previous approaches in CF explanation focus on categorical data [26, 28, 30, 32, 45], in this paper, we mainly concentrate on generating CF examples in the vision domain. One of the early approaches on counterfactual *visual* explanation is [11] where CFs are generated by exhaustively searching for feature replacement between the latent feature of query and CF images. Due to the exhaustive search for individual samples, this method is slow in practice and the generated CF images are oftentimes off the data manifold. Later, [46] proposed SCOUT in which the regions that are exclusively informative for the query or the CF classes are discovered using attribution maps. However, this work does not compose CF images and the explanations are limited to highlighting regions over images. Unlike our work, the quality of the explanations on both aforementioned approaches relies on the choice of CF images from the training set and the heuristics used for finding them.

[7] proposes a contrastive explanation framework with the goal of finding minimal and sufficient input features in order to justify the prediction or finding minimal and sufficient perturbations in order to change the classifier's prediction from the query class to a CF one (pertinent negative). Applying such perturbations is limited to gray-scale images. Although the authors suggested using an auto-encoder loss term to align CF examples to the distribution of the original data, the generated CF examples are adversarial and off the data manifold. The authors in [24] proposed to incorporate a prototype loss in the optimization of [7], making the generated CFs more interpretable. These methods usually push the generated images off the manifold of the natural images and are limited to simple datasets. We have also observed that occasionally their optimizations do not converge.

ExplainGAN [39] composes CF examples by filling a masked area over the input using a generator. Their design has an additional mask generator that needs to be trained jointly with the GAN. This is an impediment to plugging their method into existing GANs and extending the scope of their work to complicated datasets such as ImageNet.

More recently, [40] decomposes image generation into parallel mechanisms (shape, texture, and background) and the distributions over the individual mechanisms are learned. Their work generates high-resolution images, but do not explain the decision of a classifier.

[34] builds a graph over all the candidates in the training set and selected CFs from it to respect the underlying data distribution. This assumes a counterfactual example to the query image can be found among the training examples, which may not be always true. [10, 19, 37] use conditional

VAE-based architectures to generate CFs. They solve individual optimizations for each sample. Similarly, many other previous CF explanation methods [7, 11, 23, 26, 30, 38] have separate optimizations for each query image. This obstructs their applications in real-time. In turn, our method learns a transformation from query to CF (and vice versa) over the course of training. At inference time, there is no optimization to be solved and our method is suitable for interactive use.

[50] learns an unpaired image-to-image translation using cycle-consistent adversarial training. However, it has a different goal than ours. While our method can explain the decisions from *any* classifier, their method does not and instead uses two separate discriminators in order for the transformed images to lie in the target image class.

Latent Manipulations in Generative Models It has been shown that GANs learn interpretable directions in their latent space and meaningful changes can be obtained by steering in such directions. [15] shows that by linearly walking in the latent space of pretrained GANs, simple edits on images (e.g. zoom, rotation, etc.) can be learned. [13, 42, 49] aim to learn interpretable directions in the latent space of the GANs for attribute manipulation such as face editing (e.g. age, expressions, etc.). However, their manipulations do not explain the decision of an external classifier. Our method, on the other hand, explains any given classifier using the same GAN/VAE backbone.

3. Methodology

3.1. Generating CFs by Transformation in the Latent Space

As some prior work [37, 39], we utilize a generative model in order to obtain more realistic counterfactual examples that stay close to the data manifold. Toward that goal, we follow the recent idea of steerability in the latent space of generative models [15] and propose to *learn* a transformation in the latent space to obtain the CFs. Given a pretrained classifier f that we are attempting to explain, a pretrained generator G, an input (query) image $\boldsymbol{x} \in \mathbb{X}_c$ from the images in the training set with the query class c, and a target CF class c', we re-define the CF generation problem [18, 45] to learn a (non-linear) transformation $g : \mathbb{R}^D \to \mathbb{R}^D$ in the latent space of the generator that maps the latent code of the input ($\boldsymbol{z}_x$) to a CF one,

$$g^* = \arg\min_g \mathbb{E}_{\boldsymbol{x}\in\mathbb{X}_c} \left[\mathcal{L}_{cls}\left(f\left(\boldsymbol{x}'\right), c'\right) + \mathcal{L}_{prx}\left(\boldsymbol{x}', \boldsymbol{x}\right)\right]$$
$$s.t. \quad \boldsymbol{x}' = G\left(g^n\left(\boldsymbol{z}_x\right)\right), \quad \boldsymbol{z}_x = E\left(\boldsymbol{x}\right) \tag{1}$$

where $\boldsymbol{x}'$ is the generated CF and $g^n(.)$ is an n-th order function decomposition $g(g(g(\ldots)))$ — g is recursively applied n times, mimicking discrete Euler ODE approximations. Here, $g(.)$ is estimated using a simple neural network. $\mathcal{L}_{cls}$ is the *classification* loss so that the generated CF $\boldsymbol{x}'$ belongs to class c' and $\mathcal{L}_{prx}$ is the *proximity* loss encouraging $\boldsymbol{x}'$ to be proximal to the input $\boldsymbol{x}$. To get to the latent code $\boldsymbol{z}_x$ from the query image $\boldsymbol{x}$, a pretrained encoder $E : \mathbb{R}^{C\times H\times W} \to \mathbb{R}^D$ can be used.

It is worth mentioning that the main difference between this formulation and prior explanation work [18, 37] is that g is a transformation that can be directly applied on any new query image once learned, whereas prior work would need to solve separate optimization problems for every new image. The difference between this formulation and a regular conditional GAN is that our approach is used to explain a generic classifier f that is independent of the GAN, whereas conditional GANs use their discriminators to encode class knowledge. In practice, a significant amount of work is put into training discriminative classifiers and it would be desirable to have a tool that can diagnose any pre-trained classifier with a joint re-training with the GAN.

Note that in our formulation E is not an integral part and $\boldsymbol{z}_x$ can be obtained by directly sampling from the latent space distribution of the generator. To put it differently, for the purpose of training g, our method does *not require* access to the images $\boldsymbol{x}$ and sampling $\boldsymbol{z}_x$ is sufficient. When directly sampling $\boldsymbol{z}_x$, the input image is $\boldsymbol{x} = G(\boldsymbol{z}_x)$. For unconditional generative models, rejection sampling needs to be used to select $\boldsymbol{z}_x$ — based on the classifier's prediction $\boldsymbol{z}_x = \{\boldsymbol{z} | c = \arg\max f(G(\boldsymbol{z}))\}$. In the case of conditional generative models, sampling $\boldsymbol{z}$ from class c is trivial. Directly sampling $\boldsymbol{z}_x$ is particularly advantageous when using GANs as obtaining the latent GAN codes from images is still an open research topic [47]. When using VAEs as the generative models, however, obtaining latent code is straightforward where E is the encoder of the VAE.

3.2. From Query to CF and Back: Cycle-Consistent CF Generation

Finding a transformation g is highly under-constrained and there might be many solutions to the optimization problem (1) that satisfy the CF properties equally. To regularize the optimization, we modify the objective and incorporate *cycle consistency* [50] between the query and CF latent codes. That can be achieved by introducing another transformation $h : \mathbb{R}^D \to \mathbb{R}^D$ that estimates the inverse of g, *i.e.*, finds a (non-linear) trajectory in the latent space that maps the CF latent code back to the query one, *i.e.*, $\boldsymbol{z}_x \approx \boldsymbol{z}_x^{cyc}$ where $\boldsymbol{z}_x^{cyc} = h^n\left(g^n\left(\boldsymbol{z}_x\right)\right)$. We also define the *cycled* query image as $\boldsymbol{x}^{cyc} = G\left(\boldsymbol{z}_x^{cyc}\right)$ and add the *cycle* loss to objective (1). Note that $\boldsymbol{x}'$ and $\boldsymbol{x}^{cyc}$ belong to two distinct classes (the CF and the query class, respectively).

Although generating CFs from the latent space of generative models helps with staying close to the data manifold, it does not guarantee such. To ensure staying on the data manifold, we add an *adversarial* loss to objective (1). More formal descriptions of the adversarial and cycle losses are

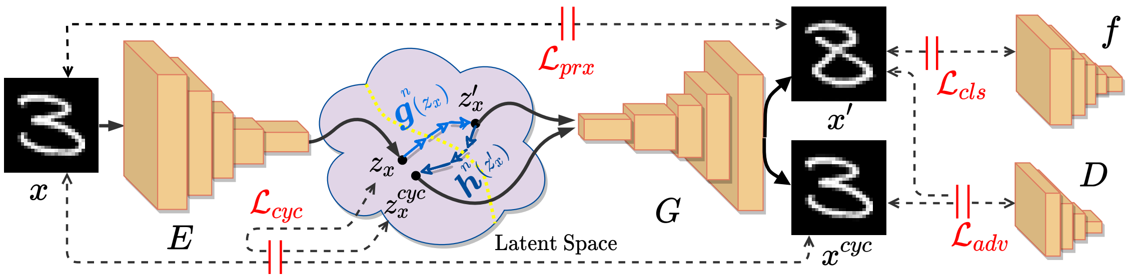

Figure 1. **Cycle-Consistent Counterfactuals by Latent Transformations (C3LT)**. This figure illustrates the architecture of our proposed framework. After a latent code $\boldsymbol{z}_x$ is obtained, our method transformed it to the CF latent code $\boldsymbol{z}'_x$ using g. The CF example can be obtained by $\boldsymbol{x}' = G\left(g^n\left(\boldsymbol{z}_x\right)\right)$. The inputs to the loss functions are outlined via dashed lines. The classifier f and discriminator D are only used during the training. (Best viewed in color)

presented later in this subsection.

Here, we formalize the main objective of our method, Cycle-Consistent Counterfactuals by Latent Transfomrations (C3LT). For cycle consistency, our method requires access to samples from both the query and the CF classes. Given an image $\boldsymbol{x} \in \mathbb{X}_c$ from the images in the training set with the query class c, an image $\boldsymbol{y} \in \mathbb{X}_{c'}$ from the CF class c', our method learns transformations g^* and h^*,

$$g^*, h^* = \arg\min_{g,h} \mathbb{E}_{\boldsymbol{x}\in\mathbb{X}_c}\left[\mathcal{L}_{c3lt}(\boldsymbol{x}, c', g, h)\right] + \mathbb{E}_{\boldsymbol{y}\in\mathbb{X}_{c'}}\left[\mathcal{L}_{c3lt}(\boldsymbol{y}, c, h, g)\right] \tag{2}$$

where,

$$\begin{aligned}\mathcal{L}_{c3lt}(\boldsymbol{x}, c', g, h) =& \mathcal{L}_{cls}\left(f\left(\boldsymbol{x}'\right), c'\right) + \mathcal{L}_{prx}\left(\boldsymbol{x}', \boldsymbol{x}\right) + \\ & \mathcal{L}_{cyc}\left(\boldsymbol{x}^{cyc}, \boldsymbol{x}\right) + \mathcal{L}_{adv}(\boldsymbol{x}', \boldsymbol{x}^{cyc}) \\ s.t. \quad & \boldsymbol{x}' = G\left(\boldsymbol{z}'_x\right), \ \boldsymbol{z}'_x = g^n\left(\boldsymbol{z}_x\right), \boldsymbol{x}^{cyc} = G\left(\boldsymbol{z}_x^{cyc}\right), \\ & \boldsymbol{z}_x^{cyc} = h^n\left(\boldsymbol{z}'_x\right), \boldsymbol{z}_x = E(\boldsymbol{x}),\end{aligned} \tag{3}$$

C3LT learns transformations between the query and CF classes at the same time, hence the query and CF notations are interchangeable. For brevity purposes, we skip the formal definition of the $\mathcal{L}_{c3lt}(\boldsymbol{y}, c, h, g)$. Fig. 1 shows the architecture of our proposed framework. In what follows, we define the individual loss terms in Eq. (3):

Classification Loss $\mathcal{L}_{cls}$ encourages the generated CF examples to be classified as the CF class. We use the Negative Log-Likelihood loss,

$$\mathcal{L}_{cls} = -\log\left(f_{c'}\left(\boldsymbol{x}'\right)\right) \tag{4}$$

where $f_{c'}\left(\boldsymbol{x}'\right)_{c'}$ is the output of the classifier for class c'.

Proximity Loss $\mathcal{L}_{prx}$ helps the generated CF examples to stay close to the query image in terms of some distance function *i.e.* CFs that are proximal to the query ones. It is also desirable that the CFs have sparse changes compared to the input images *i.e.* only a few input features change. To that end, we opt to choose an $L1$ loss term for the proximity loss. In addition, we use entropy and smoothness losses ($\mathcal{L}_{entr}$ and $\mathcal{L}_{smth}$) [39] over the absolute difference between the query and CF images to encourage changes to be more sparse and local,

$$\mathcal{L}_{prx} = ||\boldsymbol{x} - \boldsymbol{x}'||_1 + \mathcal{L}_{entr}\left(\boldsymbol{x}, \boldsymbol{x}'\right) + \mathcal{L}_{smth}\left(\boldsymbol{x}, \boldsymbol{x}'\right) \tag{5}$$

Cycle-Consistency Loss $\mathcal{L}_{cyc}$ enforces the cycle-consistency between the latent codes for the query and CF classes. In addition to the latent codes, we use Perceptual similarity [17] over the input $\boldsymbol{x}$ and the cycled image $\boldsymbol{x}^{cyc}$,

$$\begin{aligned}\mathcal{L}_{cyc} =& \sum_{l\in\mathbb{L}} ||\hat{f}^l(\boldsymbol{x}^{cyc}) - \hat{f}^l\left(\boldsymbol{x}\right)||_1 + ||\boldsymbol{x}^{cyc} - \boldsymbol{x}||_1 \\ & + ||\boldsymbol{z}_x - \boldsymbol{z}_x^{cyc}||_1 \\ s.t. \quad & \boldsymbol{z}_x^{cyc} = h^n\left(g^n\left(\boldsymbol{z}_x\right)\right), \ \boldsymbol{x}^{cyc} = G\left(\boldsymbol{z}_x^{cyc}\right)\end{aligned} \tag{6}$$

where $\hat{f}^l(\boldsymbol{x})$ is the intermediate features of a pretrained classifier $\hat{f}$ at layer l for a given input $\boldsymbol{x}$, and $\mathbb{L}$ is the set of all target layers. Note, classifier $\hat{f}$ can be the same as or different from the original classifier f that we are explaining.

Adversarial Loss $\mathcal{L}_{adv}$ helps the generated CFs and *cycled* images to lie close to the manifold of the original data using the discriminator D,

$$\mathcal{L}_{adv} = \log\left(1 - D\left(\boldsymbol{x}^{cyc}\right)\right) + \log\left(1 - D\left(\boldsymbol{x}'\right)\right) \tag{7}$$

so that $\boldsymbol{x}^{cyc}$ and $\boldsymbol{x}'$ pose to the discriminator as real images.

3.3. Inference

At the inference time, when an encoder E is available, the input $\boldsymbol{x}$ goes through the encoder to obtain the latent code $\boldsymbol{z}_x = E(\boldsymbol{x})$. It is then transformed by g^*, followed by passing through the generator G to obtain the CF example $\boldsymbol{x}' = G\left(\left(g^*\right)^n\left(\boldsymbol{z}_x\right)\right)$. This results in *fast* inference and makes our method suitable for interactive applications — unlike many of the previous approaches [7,11,18,23,26,30,37,38] where the CFs are generated by solving optimization problem for individual inputs.

When an encoder E is not available, the inference is slightly different; given an input image $\boldsymbol{x}$, the latent code is calculated by $\boldsymbol{z}_x^* = \arg\min_{\boldsymbol{z}} \mathcal{L}(G(\boldsymbol{z}), \boldsymbol{x})$ [12,47] which is slower than when the encoder is available. The rest is similar to the above procedure. Note that when there are no input images, the query and CF classes can be inspected by sampling $\boldsymbol{z}_x$ directly from the latent space distribution.

4. Experiments

4.1. Setup

MNIST and Fashion-MNIST. We evaluate the C3LT method against CF explanation baselines, namely, Contrastive Explanation Method (CEM) [7], Counterfactual Visual Explanation (CVE) [11], and ExplainGAN (ExpGAN) [39] on the MNIST [22] and Fashion-MNIST [48] datasets by both qualitative inspection and an extensive set of quantitative metrics. Due to the similarities in CF explanation and adversarial attacks, we also generate adversarial examples on the query images using the PGD attack (denoted as Adv. Attack) [25] with the CF class as the target. Images from both datasets have 28×28 resolutions and 10 classes. We use the standard train/test split. The C3LT and [39] use the examples from the query and CF classes in the train set ($\sim$6,000 samples/class) for the purpose of training and the examples from the query/CF class in the test set ($\sim$1,000 samples/class) for evaluation. While we used official implementations for [7, 11], we could not find any available implementations for [39] and implemented it ourselves.

[7, 11] directly evaluate on the test set since they solve optimization problems for individual samples without any training. In addition, given an input $\boldsymbol{x}$ from class c, [7, 11] do not take user-specified CF class c' as the target and just aim to change the classifier's output to the *maximum-non-query* class $\arg\max_{i \neq c} f(\boldsymbol{x})$. To have a fair comparison across all baselines, however, we slightly modify their objective and instead select the CF class c' as the target.

Similar to [39], for the MNIST dataset, we use query and CF class pairs (3, 8), (4, 9), and (5, 6). For Fashion-MNIST, we use (coat, shirt), (t-shirt, pullover), and (sneaker, boot). Unlike C3LT and [39], [7, 11] do not guarantee a solution, and we found their optimizations occasionally do not converge hence would not be able to find any CF explanations. For a fair comparison, we only consider the samples that all methods successfully generated CF explanation for. The reported numbers in the following sections are averaged over all samples and pairs for each dataset. Across all methods, we use the same classifier f for explaining. The architecture of the classifier used is the same for both datasets where it obtains 99.4% and 91.5% test set accuracy on MNIST and Fashion-MNIST datasets, respectively.

Regarding C3LT, we train an encoder E to map the input images to the corresponding code in the latent space of the generator. For the choice of the generator G, we used a pre-trained DC-GAN [35] and PGAN [20] for MNIST and Fashion-MNIST, respectively. We use similar discriminators as used by DCGAN method. Moreover, we used a simple 2-layer fully-connected neural network with ReLU activation for the choice of transformations g and h.

ImageNet from BigGAN. To showcase the capability of our framework in generating CFs on high-resolution real-world data, we use C3LT on ImageNet [6]-trained BigGAN [4], a conditional GAN that generates high-fidelity and high-quality images. To the best of our knowledge, the C3LT is the first CF explanation method to generate CFs that explain classifiers for high-resolution natural images such as ImageNet. This is possible due to the flexibility of our framework and its modular nature. Here, we sample $\boldsymbol{x}_x$ directly from the latent space distribution $\boldsymbol{z}_x \sim \mathcal{N}(0, I)$ with truncation 0.4 and an encoder E is not required. We opt to use pre-trained *BigGAN-deep* at 256×256 resolution. We use the (*leopard, tiger*), (*Egyptian cat, Persian cat*), (*rooster, hen*), (*husky, wolf*), and (*pembroke corgi, cardigan corgi*) class pairs. The experiments on BigGAN are limited to C3LT (our method) as baselines are incompetent in generating meaningful CFs and we do not quantitatively compare against them. For further details on the experiments, please refer to the supplementary materials A.

4.2. Qualitative Inspection of the Counterfactuals

Fig. 2 shows some examples of generated CFs for high-resolution images using BigGAN. This showcases that our method can be plugged into state-of-the-art GAN models and generate CFs by finding transformations in their latent space. It can be observed that our method pays attention to both *foreground* and *background* in the image and mainly keeps the background the same. C3LT found transformations in both the *global shape* and the *texture* of the objects according to their category. For instance, for going from rooster to hen, the shape and texture are changed, e.g., smaller legs, smaller comb, and bigger belly, and the texture of the breast is slightly altered. For the leopard to tiger and corgi examples, however, the main transformations are occurring in the texture and color of the objects.

In Fig. 3, we present the CF examples obtained from C3LT and other baselines from the MNIST and Fashion-MNIST datasets. We show comparison across all the pairs used in the evaluation. We found ExpGAN to generate more interpretable explanations than other baselines. However, the CFs are often un-natural with diffuse perturbations (e.g. images 4 and *t-shirt*). We suspect this is due to their CF composition mechanism using a mask over the input. Further, we found the CFs to be occasionally adversarial where the mask generation fails (e.g. images 3 and *coat*). Not to our surprise, the CFs obtained from the CEM were mostly adversarial and the perturbations were hardly perceptible (e.g. images 5 and *coat*, *sneaker*). Although replacement of patches of pixels and losing the global shape might fool the CNN [3], we did not find the CFs from the CVE to be interpretable (e.g. images 5 and *coat*). Inspecting through the generated CFs, we found our method to consistently generate interpretable and realistic images. Quantitative results

Figure 2. **High-resolution** (256×256) **Counterfactual Generation**. This figure shows high-resolution CF explanations generated using C3LT for (*leopard, tiger*), (*Egyptian cat, Persian cat*), (*rooster, hen*), (*husky, wolf*), and (*pembroke corgi, cardigan corgi*) CF pairs, respectively from left to right.

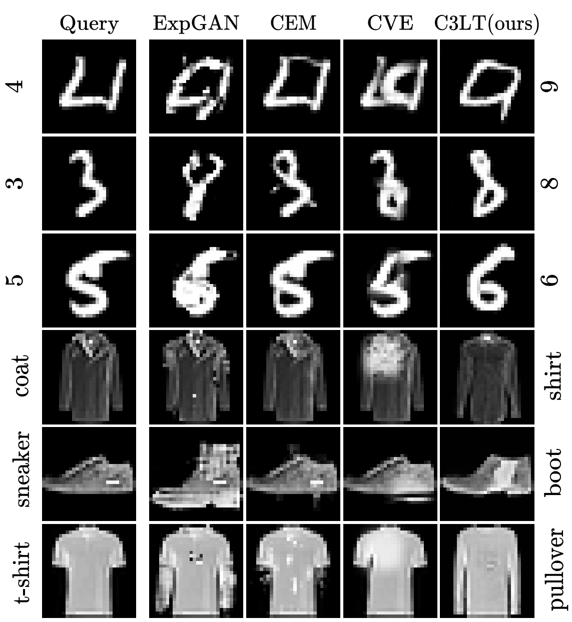

Figure 3. **Qualitative Comparison of the CFs**. This figure shows CF explanations obtained from our method (C3LT) and baselines across MNIST and Fashion-MNIST datasets. Broadly, we find the generated CFs from the CEM and CVE to be adversarial and off the data manifold. Compared to the ExpGAN, the generated CFs from our method are consistently more realistic and interpretable.

obtained from section 4.3.2 support our findings.

4.2.1 Can C3LT Generate CFs on Nonsimilar Classes?

Although the CF classes in the experiments were chosen to be close to the query one, C3LT, as a method, generates CF examples for any class pairs. To show this, we generate CFs for non-similar class pairs (3, 4) from MNIST and (boot, pullover) from Fashion-MNIST in Fig. 4. Note that although the changes are significant, certain latent attributes that are independent of the category are still preserved, e.g., in the pairs between handwritten 3s and 4s, the stroke width and writing style are preserved, and in the boots/pullover case, one can see that slim clothes transfer to slim boots and vice versa.

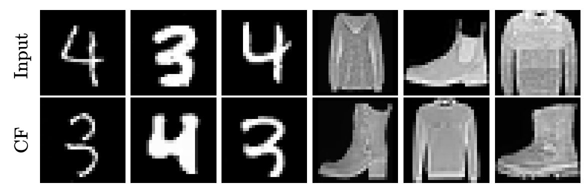

Figure 4. CF examples on non-similar classes using C3LT.

4.3. Quantitative Evaluation of the CF Explanations

4.3.1 Counterfactual Transition Metric

One of the main challenges in explaining deep networks is defining automatic metrics for quantitative evaluation. The authors in ExpGAN [39] treat the mask generated from their method as a pixel-wise attribution map and evaluate it against attribution map baselines. However, we argue this would be a relevant comparison for *factual* explanations rather than for CF ones. This is mainly due to the fact that in the evaluation of attribution maps, only changes in the output score for the query class are considered while changes in the output score of the CF class are dismissed. In the following, we propose a new metric called *COUnterfactual Transition (COUT)* metric to address this shortcoming.

Inspired from the *deletion* metric [33], we devise a new metric to consider the changes in the output of the classifier for the query and the CF classes simultaneously, making it suitable for automatic evaluation of CF explanation methods. Given a query image $\boldsymbol{x}$, a generated CF example $\boldsymbol{x}'$, and a $mask \in [0, 1]$ indicating the spatial location and *relative* amount of changes over the query image

Table 1. **COUnterfactual Transition (COUT) Scores**. The COUT metric measures the quality of the generated CFs and implicitly evaluates their validity and sparsity properties (I,III), *i.e.*, it favors *small* amount of changes that *maximizes* the output score for the CF class (minimizes the output score for the query class). COUT is computed as the difference between the area under the curve for the classifier's output of both the query ($AUPC_c$) and CF ($AUPC_{c'}$) classes. The higher the COUT score the better.

Methods	ExpGAN [39]		CEM [7]		CVE [11]		C3LT (ours)		Adv. Attack [25]	
	Mnist	FMnist	Mnist	FMnist	Mnist	FMnist	Mnist	FMnist	Mnist	FMnist
$AUPC_{c'} \uparrow$	0.967	0.920	0.301	0.347	0.209	0.275	**0.980**	**0.958**	0.737	0.732
$AUPC_c \downarrow$	0.040	0.062	0.555	0.427	0.767	0.638	**0.031**	**0.052**	0.266	0.481
$COUT \uparrow$	0.927	0.858	-0.253	-0.080	-0.557	-0.363	**0.948**	**0.906**	0.471	0.251

needed to get to the CF one, the COUT metric is calculated as following; first, the pixel values in the (normalized) mask are sorted based on their values. Next, for a fix number of steps T, batches of pixels are inserted from the CF example into the query one according to the ordered masks values. The changes in the output score of the classifier for *both* the query c and CF c' classes are measured. The Area Under the Perturbation Curve (*AUPC* $\in [0,1]$) for each class $k \in \{c, c'\}$ is then calculated. Their difference is reported as the COUT $\in [-1,1]$ score,

$$\text{COUT} = AUPC_{c'} - AUPC_c \tag{8}$$

$$AUPC_k = \frac{1}{T}\left\langle \sum_{t=0}^{T-1} \frac{1}{2}\left(f_k\left(\boldsymbol{x}^{(t)}\right) + f_k\left(\boldsymbol{x}^{(t+1)}\right)\right)\right\rangle_{p_{\text{data}}}$$

where $\boldsymbol{x}^{(t)}$ is the input after $t \in \{0, \ldots, T\}$ steps perturbations while $\boldsymbol{x}^{(0)}$ being the query image ($\boldsymbol{x} = \boldsymbol{x}^{(0)}$) and the $\boldsymbol{x}^{(T)}$ the CF one ($\boldsymbol{x}' = \boldsymbol{x}^{(T)}$) (see Fig. 5), $f_k()$ is the classifier's output for class k, and $< . >_{p_{\text{data}}}$ denotes the average over all images in the evaluation data. Some methods such as ExpGAN explicitly generate the $mask$. However, for the rest of the baselines, given the CF and query images, it can be obtained by calculating the absolute difference between the images and normalizing it between 0 and 1.

The COUT metric measures the amount of change that would be needed to move a query image into the CF class. Besides the classification change, it measures how fast the output score for the CF class maximizes, and in an opposite way, the output score for the query class minimizes. This favors methods that find sparse changes over the input features that crucially shift the output of the classifier from the query class to the CF one. In other words, COUT measures both properties I and III as defined in the introduction.

Table 1 summarizes the COUT results obtained from our method and the baselines. CVE is generating the CF examples by a few discrete edits. For a fair comparison, we calculate the COUT metric for CVE slightly differently where we calculate the AUPC by measuring the output score after each edit. Both CEM and CVE perform poorly on the COUT metric as their optimization does not reach a high output score for the CF class. This also results in low validity of their generated CFs (see section 4.3.3). Our method consistently outperforms the baselines in terms of the $AUPC_c$, $AUPC_{c'}$, and COUT on both MNIST and Fashion-MNIST datasets.

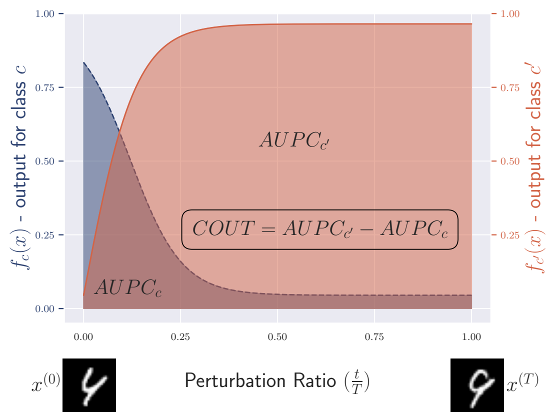

Figure 5. **COUnterfactual Transition (COUT) metric**. Here, we illustrate how the COUT metric is calculated. Starting from the query image $x^{(0)}$ (e.g. digit 4), after T steps perturbations we get to the generated CF image $x^{(T)}$ (e.g. digit 9). The area under the curve for the classifier's output of both the query ($AUPC_c$) and CF ($AUPC_{c'}$) classes is calculated — averaged over all evaluation data. The COUT metric is simply their difference.

4.3.2 Realism of the CFs

In order for the generated CF examples to be *relevant* as a means of explanation, they should have high *realism* (property IV), *i.e.*, lie close to the data manifold of the CF class. As mentioned earlier, this is one of the main challenges for CF explanations, particularly in high-dimensional input spaces such as natural images where the pitfall of adversarial solutions becomes more prominent. To this end, we evaluate the generated CF examples from our method against the baselines in terms of their realism and how well they match the distribution of the original data.

[24] proposed *IM1* and *IM2* metrics that use reconstruction errors from pre-trained auto-encoders over the images from the query, CF, and all classes to assess how well the distribution of the generated CFs match the original data. A lower *IM1* metric implies the CFs lie closer to the data manifold of the CF class rather than the query one. A lower *IM2*, on the other hand, implies the distribution of the CFs is similar to the distribution of original data from all classes. In addition, we use Fréchet Inception Distance (FID) [14] and Kernel Inception Distance (KID) [2] metrics that are well-established for evaluating the quality of the synthesized im-

Table 2. **Realism Comparison of the CFs**. In this table, we compare the generated CFs obtained from C3LT (our method) and its baselines in terms of realism, *i.e.*, how close the generated CFs lie to the data manifold. Across all metrics, C3LT outperforms the baselines and generates high-quality CFs.

Methods	ExpGAN [39]		CEM [7]		CVE [11]		C3LT (ours)		Original Data		Adv. Attack [25]	
	Mnist	FMnist	Mnist	FMnist	Mnist	FMnist	Mnist	FMnist	Mnist	FMnist	Mnist	FMnist
$IM1 \downarrow$	0.72	0.77	1.68	1.63	1.44	1.24	**0.70**	**0.74**	0.47	0.58	1.02	1.34
$IM2 \times 10 \downarrow$	0.43	0.14	1.08	0.26	1.38	0.37	**0.36**	**0.093**	0.29	0.12	2.10	0.365
$FID \downarrow$	41.12	76.52	50.03	96.87	47.53	83.77	**22.83**	**62.31**	8.37	16.26	203.07	140.28
$KID \times$ 1e3 $\downarrow$	37.27	70.44	44.88	91.71	37.24	72.71	**13.39**	**52.71**	0.34	0.03	283.50	157.27

Table 3. **Proximity and Validity**. This table compares the CF explanations methods in terms of validity and proximity. Methods that obtain high validity and low proximity are desirable.

Methods	ExpGAN [39]		CEM [7]		CVE [11]		C3LT (ours)		Adv. Attack [25]	
	Mnist	FMnist	Mnist	FMnist	Mnist	FMnist	Mnist	FMnist	Mnist	FMnist
$Prox \downarrow$	0.074	0.135	**0.016**	**0.013**	0.055	0.054	0.072	0.116	0.229	0.196
$Val \uparrow$	0.997	0.998	0.469	0.620	0.231	0.145	**0.999**	**1.0**	0.998	1.0

ages from generative models. For all of the metrics, the lower the score, the better.

Table 2 compares the CF explanation methods in terms of their realism. To have reference values for the aforementioned metrics, we also use images in the evaluation set from the CF class (*Original Data*) as a baseline. Similar to our findings in visual comparison of the CFs, our method generates more realistic and higher quality images compared to the baselines. We also find $IM1$ and $IM2$ metrics to be more close to our visual inspection of the generated CFs where methods such as CEM, CVE, and Adv. Attacks perform very poorly. However, in terms of the FID and KID metrics, their performance is relatively better, particularly for the Fashion-MNIST dataset.

4.3.3 Validity

When explaining the classifier f using CF examples, the generated CFs are expected to lie within the decision boundaries of the CF class c', *i.e.*, be valid. In addition to COUT and in order to have a simple and intuitive metric to measure the validity of CFs (property I), we define the Val metric,

$$Val = \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}_{f(\boldsymbol{x}'_n),c'} \tag{9}$$

where $\mathbb{1}_{f(\boldsymbol{x}'_n),c'}$ is the indicator function that the prediction of the classifier f for the n-th CF example $\boldsymbol{x}'_n$ is c', and N is the total number of CFs. This measures the fraction of the generated CF examples that are correctly predicted by the classifier f to the CF class c'. Table. 3 shows the obtained results from our method and compares it against baselines. Our method achieves very high Val on both MNIST and Fashion-MNIST datasets. However, [7, 11] struggle to generate valid examples, hence their explanations have low *faithfulness*. High Val scores obtained from the Adv. Attack also point out that sole reliance on the validity for CF evaluation can be misleading and it should always be considered along with other evaluation criteria such as COUT or realism (see section 4.3.2).

4.3.4 Proximity

In order to generate CF examples, minimal changes to the features of the query image are favorable (property II). We simply define the *proximity* metric as the mean of feature-wise $L1$ distances between the query and CF examples,

$$Prox = \sum_{n=1}^{N} \frac{||\boldsymbol{x}_n - \boldsymbol{x}'_n||_1}{NCHW} \tag{10}$$

where $\boldsymbol{x}_n$ and $\boldsymbol{x}'_n$ are the n-th query and CF example from the evaluation set, and C, H and W are the number of channels, height, and width of the input image, respectively. Table 3 compares our method against the baselines in terms of proximity of the generated CFs. It can be seen that [7, 11] outperform other methods on this metric. However, as mentioned earlier, the generated CFs from their method do not have high realism (see section 4.3.2) and faithfulness (see section 4.3.3) hence not reliable.

5. Conclusion

In this paper, we presented a novel framework for generating counterfactual explanations by learning a transformation function in the latent space of a generative model (GAN/VAE) with a combination of several loss functions including cycle-consistency. Extensive experiments show that our approach outperforms prior work across all metrics, as well as possessing two desirable properties: *First*, it does not require joint training with a generative model hence can be plugged into state-of-the-art generative algorithms to generate high-resolution CFs. *Second*, once learned, our approach can generate CF examples on-the-fly during inference time which makes it ideal to be used in practical systems to explain deep networks.

Acknowledgements

This work is supported in part by DARPA contract N66001-17-2-4030. We also thank Dr. Xin Wang from the Samaritan Albany General Hospital who inspired us to perform this research.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 11

[2] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. In *International Conference on Learning Representations*, 2018. 7

[3] Wieland Brendel and Matthias Bethge. Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. In *International Conference on Learning Representations*, 2018. 5

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018. 5

[5] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: Deep learning for interpretable image recognition. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 1

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5

[7] 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 *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 590–601, 2018. 1, 2, 3, 4, 5, 7, 8, 11, 12

[8] Nicholas Frosst and Geoffrey Hinton. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*, 2017. 1

[9] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. *Advances in Neural Information Processing Systems*, 32:9277–9286, 2019. 1

[10] Yash Goyal, Amir Feder, Uri Shalit, and Been Kim. Explaining classifiers with causal concept effect (cace). *arXiv preprint arXiv:1907.07165*, 2019. 2

[11] Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In *International Conference on Machine Learning*, pages 2376–2384. PMLR, 2019. 1, 2, 3, 4, 5, 7, 8, 11, 12

[12] Jinjin Gu, Yujun Shen, and Bolei Zhou. Image processing using multi-code gan prior. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3012–3021, 2020. 4

[13] Shuyang Gu, Jianmin Bao, Hao Yang, Dong Chen, Fang Wen, and Lu Yuan. Mask-guided portrait editing with conditional gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3436–3445, 2019. 3

[14] 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. *Advances in neural information processing systems*, 30, 2017. 7

[15] Ali Jahanian, Lucy Chai, and Phillip Isola. On the" steerability" of generative adversarial networks. In *International Conference on Learning Representations*, 2019. 3

[16] Jeya Vikranth Jeyakumar, Joseph Noor, Yu-Hsi Cheng, Luis Garcia, and Mani Srivastava. How can i explain this to you? an empirical study of deep neural network explanation methods. *Advances in Neural Information Processing Systems*, 2020. 1

[17] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016. 4

[18] Shalmali Joshi, Oluwasanmi Koyejo, Been Kim, and Joydeep Ghosh. xgems: Generating examplars to explain black-box models. *arXiv preprint arXiv:1806.08867*, 2018. 3, 4

[19] Shalmali Joshi, Oluwasanmi Koyejo, Warut Vijitbenjaronk, Been Kim, and Joydeep Ghosh. Towards realistic individual recourse and actionable explanations in black-box decision making systems. *arXiv preprint arXiv:1907.09615*, 2019. 2

[20] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. 5

[21] Saeed Khorram, Tyler Lawson, and Li Fuxin. igos++ integrated gradient optimized saliency by bilateral perturbations. In *Proceedings of the Conference on Health, Inference, and Learning*, pages 174–182, 2021. 1

[22] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5

[23] Shusen Liu, Bhavya Kailkhura, Donald Loveland, and Yong Han. Generative counterfactual introspection for explainable deep learning. In *2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 1–5. IEEE, 2019. 2, 3, 4

[24] Arnaud Van Looveren and Janis Klaise. Interpretable counterfactual explanations guided by prototypes. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 650–665. Springer, 2021. 2, 7

[25] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. 5, 7, 8, 11

[26] Divyat Mahajan, Chenhao Tan, and Amit Sharma. Preserving causal constraints in counterfactual explanations for machine learning classifiers. *arXiv preprint arXiv:1912.03277*, 2019. 2, 3, 4

[27] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019. 1

[28] Jonathan Moore, Nils Hammerla, and Chris Watkins. Explaining deep learning models with constrained adversarial examples. In *Pacific Rim International Conference on Artificial Intelligence*, pages 43–56. Springer, 2019. 1, 2

[29] Raha Moraffah, Mansooreh Karami, Ruocheng Guo, Adrienne Raglin, and Huan Liu. Causal interpretability for machine learning-problems, methods and evaluation. *ACM SIGKDD Explorations Newsletter*, 22(1):18–33, 2020. 2

[30] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 607–617, 2020. 1, 2, 3, 4

[31] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 11

[32] Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. Learning model-agnostic counterfactual explanations for tabular data. In *Proceedings of The Web Conference 2020*, pages 3126–3132, 2020. 2

[33] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. In *British Machine Vision Conference (BMVC)*, 2018. 6

[34] Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. Face: Feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 344–350, 2020. 2

[35] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 5

[36] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016. 1

[37] Pau Rodríguez, Massimo Caccia, Alexandre Lacoste, Lee Zamparo, Issam Laradji, Laurent Charlin, and David Vazquez. Beyond trivial counterfactual explanations with diverse valuable explanations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1056–1065, October 2021. 1, 2, 3, 4

[38] Chris Russell. Efficient search for diverse coherent explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 20–28, 2019. 2, 3, 4

[39] Pouya Samangouei, Ardavan Saeedi, Liam Nakagawa, and Nathan Silberman. Explaingan: Model explanation via decision boundary crossing transformations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 666–681, 2018. 1, 2, 3, 4, 5, 6, 7, 8, 11, 12

[40] Axel Sauer and Andreas Geiger. Counterfactual generative networks. In *International Conference on Learning Representations*, 2020. 2

[41] 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. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 1

[42] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *CVPR*, 2020. 3

[43] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. 1

[44] Sahil Verma, John Dickerson, and Keegan Hines. Counterfactual explanations for machine learning: A review. *arXiv preprint arXiv:2010.10596*, 2020. 2

[45] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017. 1, 2, 3

[46] Pei Wang and Nuno Vasconcelos. Scout: Self-aware discriminant counterfactual explanations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8981–8990, 2020. 1, 2

[47] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. Gan inversion: A survey. *arXiv preprint arXiv:2101.05278*, 2021. 3, 4

[48] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. 5

[49] Huiting Yang, Liangyu Chai, Qiang Wen, Shuang Zhao, Zixun Sun, and Shengfeng He. Discovering interpretable latent space directions of gans beyond binary attributes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12177–12185, 2021. 3

[50] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 2, 3, 11, 12, 13