

Beyond Trivial Counterfactual Explanations with Diverse Valuable Explanations

Pau Rodríguez¹ Massimo Caccia^{1,2,3} Alexandre Lacoste¹ Lee Zamparo¹ Issam Laradji^{1,2,4}

Laurent Charlin^{2,5,6} David Vazquez¹

¹Element AI ²MILA ³UdeM ⁴McGill University ⁵HEC Montreal ⁶CIFAR AI Chair

pau.rodriguez@servicenow.com

Abstract

Explainability for machine learning models has gained considerable attention within the research community given the importance of deploying more reliable machine-learning systems. In computer vision applications, generative counterfactual methods indicate how to perturb a model’s input to change its prediction, providing details about the model’s decision-making. Current methods tend to generate trivial counterfactuals about a model’s decisions, as they often suggest to exaggerate or remove the presence of the attribute being classified. For the machine learning practitioner, these types of counterfactuals offer little value, since they provide no new information about undesired model or data biases. In this work, we identify the problem of trivial counterfactual generation and we propose DiVE to alleviate it. DiVE learns a perturbation in a disentangled latent space that is constrained using a diversity-enforcing loss to uncover multiple valuable explanations about the model’s prediction. Further, we introduce a mechanism to prevent the model from producing trivial explanations. Experiments on CelebA and Symbols demonstrate that our model improves the success rate of producing high-quality valuable explanations when compared to previous state-of-the-art methods. Code is available at <https://github.com/ElementAI/beyond-trivial-explanations>.

1. Introduction

Consider an image recognition model such as a smile classifier. In case of erroneous prediction, an explainability system should provide information to machine learning practitioners to understand why such error happened and how to prevent it. Counterfactual explanation methods [4, 11, 13] can help highlight the limitations of an ML model by uncovering data and model biases. Counterfactual explanations provide perturbed versions of the input data that emphasize features that contributed the most to the ML model’s output. For the smile classifier, if the

model is confused by people wearing sunglasses then the system could generate alternative images of faces without sunglasses that would be correctly recognized. In order to discover a model’s limitations, counterfactual generation systems could be used to generate images that would confuse the classifier, such as people wearing sunglasses or scarfs occluding the mouth. This is different from other types of explainability methods such as feature importance methods [4, 54, 55] and boundary approximation methods [40, 51], which highlight salient regions of the input like the sunglasses but do not indicate how the ML model could achieve a different prediction.

According to [41, 52], counterfactual explanations should be *valid*, *proximal*, and *sparse*. A *valid* counterfactual explanation changes the prediction of the ML model, for instance, adding sunglasses to confuse a smile classifier. The explanation is *sparse* if it only changes a minimal set of attributes, for instance, it only adds sunglasses and it does not add a hat, a beard, or the like. An explanation is *proximal* if it is perceptually similar to the original image, for instance, a ninety degree rotation of an image would be a sparse but not proximal. In addition to the three former properties, generating a set of *diverse* explanations increases the likelihood of finding a useful explanation [41, 52]. A set of counterfactuals is diverse if each one proposes to change a different set of attributes. Following the previous example, a diverse set of explanations would suggest to add or remove sunglasses, beard, or scarf, while a non-diverse set would all suggest to add or remove different brands of sunglasses. Intuitively, each explanation should shed light on a different action that a user can take to change the ML model’s outcome.

Current generative counterfactual methods like xGEM [26] generate a single explanation that is not constrained to be *similar* to the input. Thus, they fail to be *proximal*, *sparse*, and *diverse*. Progressive Exaggeration (PE) [57] provides higher-quality explanations that are more *proximal* than xGEM, but it still fails to provide a *diverse* set of explanations. In addition, the image generator of PE is trained on the same data as the image classifier in

order to detect biases thereby limiting their applicability. Both of these two methods tend to produce *trivial* explanations, which only address the attribute that was intended to be classified, without further exploring failure cases due to biases in the data or spurious correlations. For instance, an explanation that suggests to increase the ‘smile’ attribute of a ‘smile’ classifier for an already-smiling face is trivial and it does not explain why a misclassification occurred. On the other hand, a non-trivial explanation that suggests to change the facial skin color would uncover a racial bias in the data that should be addressed by the ML practitioner. In this work, we focus on *diverse valuable* explanations, that is, *valid*, *proximal*, *sparse*, and *non-trivial*.

We propose Diverse Valuable Explanations (DiVE), an explainability method that can interpret ML model predictions by identifying sets of valuable attributes that have the most effect on model output. In order to generate *non-trivial* explanations, DiVE leverages the Fisher information matrix of its latent space to focus its search on the less influential factors of variation of the ML model. This mechanism enables the discovery of spurious correlations learned by the ML model. DiVE produces multiple counterfactual explanations which are enforced to be *valuable*, and *diverse*, resulting in more informative explanations for machine learning practitioners than competing methods in the literature. Our method first learns a generative model of the data using a β -TCVAE [5] to obtain a disentangled latent representation which leads to more *proximal* and *sparse* explanations. In addition, the VAE is not required to be trained on the same dataset as the ML model to be explained. DiVE then learns a latent perturbation using constraints to enforce *diversity*, *sparsity*, and *proximity*.

We provide experiments to quantify the success of explainability systems at finding *valuable* explanations. We find that DiVE is more successful at finding *non-trivial* explanations than previous methods and baselines. In addition, we provide experiments to compare the quality of the generated explanations with the current state-of-the-art. First, we assess their *validity* on the CelebA dataset [36] and provide quantitative and qualitative results on a bias detection benchmark [57]. Second, we show that the generated explanations are more *proximal* in terms of Fréchet Inception Distance (FID) [20], which is a measure of similarity between two datasets of images commonly used to evaluate the quality of generated images. In addition, we evaluate the *proximity* in latent space and face verification accuracy, as reported by Singla et al. [57]. Third, we assess the *sparsity* of the generated counterfactuals by computing the average change in facial attributes.

We summarize the contributions of this work as follows: 1) We identify the importance of finding *non-trivial* explanations and we propose a new benchmark to evaluate how *valuable* the explanations are. 2) We propose DiVE, an ex-

plainability method that can interpret an ML model by identifying the attributes that have the most effect on its output. 3) We propose to leverage the Fisher information matrix of the latent space for finding spurious features that produce *non-trivial* explanations. 4) DiVE achieves state of the art in terms of the *validity*, *proximity*, and *sparsity* of its explanations, detecting biases on the datasets, and producing multiple explanations for an image.

2. Related Work

Explainable artificial intelligence (XAI) is a suite of techniques developed to make either the construction or interpretation of model decisions more accessible and meaningful. Broadly speaking, there are two branches of work in XAI, ad-hoc and post-hoc. Ad-hoc methods focus on making models interpretable, by imbuing model components or parameters with interpretations that are rooted in the data themselves [25, 42, 49]. To date, most successful machine learning methods, including deep learning ones, are uninterpretable [6, 18, 24, 34].

Post-hoc methods aim to explain the decisions of uninterpretable models. These methods can be categorized as non-generative and generative. Non-generative methods use information from an ML model to identify the features most responsible for an outcome for a given input. Approaches like [40, 44, 51] interpret ML model decisions by fitting a locally interpretable model. Others use the gradient of the ML model parameters to perform feature attribution [1, 54–56, 58, 64, 65], sometimes by employing a reference distribution for the features [11, 55]. This has the advantage of identifying alternative feature values that, when substituted for the observed values, would result in a different outcome. These methods are limited to small contiguous regions of features with high influence on the target model outcome. In so doing, they can struggle to provide plausible changes of the input that are useful for a user in order to correct a certain output or bias of the model. Generative methods such as [4, 5, 7, 15] propose *proximal* modifications of the input that change the model decision. However the generated perturbations are usually performed in pixel space and bound to masking small regions of the image without necessarily having a semantic meaning. Closest to our work are generative counterfactual explanation methods which synthesize perturbed versions of observed data that result in a change of the model prediction. These can be further subdivided into two families. The first family of methods conditions the generative model on attributes, by e.g. using a conditional GAN [26, 35, 53, 62, 63]. This dependency on attribute information can restrict the applicability of these methods in scenarios where annotations are scarce. Methods in the second family use generative models such as VAEs [30] or unconditional GANs [14] that do not depend on attributes during generation [9, 45, 57]. While

these methods provide *valid* and *proximal* explanations for a model outcome, they fail to provide a *diverse* set of *sparse*, *non-trivial* explanations. Mothilal et al. [41] addressed the diversity problem by introducing a diversity constraint between randomly initialized counterfactuals (DICE). However, DICE shares the same problems as [4, 7] since perturbations are directly performed on the observed feature space and it is not designed to generate *non-trivial* explanations.

In this work we note that existing counterfactual generation methods tend to produce explanations that exaggerate or reduce the main attribute being classified, a property we call trivial explanation, and propose DiVE, a counterfactual explanation method that focuses on generating *non-trivial* explanations, which change the outcome of a classifier by modifying other attributes in the images, revealing spurious correlations or biases of the classifiers to ML practitioners. We provide a more exhaustive review of the related work in the Supplementary Material.

3. Proposed Method

We propose DiVE, an explainability method that can interpret an ML model by identifying the latent attributes that have the most effect on its output. Summarized in Figure 1a, DiVE uses an encoder, a decoder, and a fixed-weight ML model for which we have access to its gradients. In this work, we focus on a binary image classifier in order to produce visual explanations. DiVE consists of two main steps. First, the encoder and the decoder are trained in an unsupervised manner to approximate the data distribution on which the ML model was trained. Unlike PE [57], our encoder-decoder model does not need to train on the same dataset that the ML model was trained on. Second, we optimize a set of vectors ϵ_i to perturb the latent representation \mathbf{z} generated by the trained encoder. The details of the optimization procedure are provided in the Supplementary Material. We use the following 3 main losses for this optimization: a counterfactual loss \mathcal{L}_{CF} that attempts to fool the ML model, a proximity loss $\mathcal{L}_{\text{prox}}$ that constrains the explanations with respect to the number of changing attributes, and a diversity loss \mathcal{L}_{div} that enforces the explainer to generate diverse explanations with only one confounding factor for each of them. Finally, we propose several strategies to mask subsets of dimensions in the latent space to prevent the explainer from producing trivial explanations. Next we explain the methodology in more detail.

3.1. Obtaining a counterfactual representation.

Given a data sample $\mathbf{x} \in \mathcal{X}$, its corresponding target $y \in \{0, 1\}$, and a potentially biased ML model $f(\mathbf{x})$ that approximates $p(y|\mathbf{x})$, our method finds a perturbed version of the same input $\tilde{\mathbf{x}}$ that produces a desired probabilistic outcome $\tilde{y} \in [0, 1]$, so that $f(\tilde{\mathbf{x}}) = \tilde{y}$. In order to produce semantically meaningful counterfactual explanations, we seek

to learn a counterfactual model of the image generator with the corresponding latent representation $\mathbf{z} \in \mathcal{Z} \subseteq \mathbb{R}^d$ of the input \mathbf{x} . Ideally, each dimension in \mathcal{Z} represents a different semantic concept of the data, *i.e.*, the different dimensions are *disentangled*.

We note that performing counterfactual transformation on images is an unsolved problems with many challenges. Despite this, we move forward with a practical approach and verify empirically that the result is reasonable. Our general approach is inspired from Pawlowski et al. [46]. They show that when the causal graph is specified, it is possible to use a VAE to approximate counterfactual inference. In our case, we make the assumption that the underlying causal graph is a factorial z causing the image x . However, z is unobserved and cannot be identified in the general case [37]. Hence, we rely on β -TCVAE [5] with inductive bias to estimate a disentangle representation, which was shown to obtain competitive disentanglement in practice [37]. It follows the same encoder-decoder structure as the VAE [30], *i.e.*, the input data is first encoded by a neural network $q_\phi(z|\mathbf{x})$ parameterized by ϕ . Then, the input data is recovered by a decoder neural network $p_\theta(\mathbf{x}|z)$, parameterized by θ .

In addition to the β -TCVAE loss, we use the perceptual reconstruction loss from Hou et al. [21]. This replaces the pixel-wise reconstruction loss by a perceptual reconstruction loss, using the hidden representation of a pre-trained neural network R . Specifically, we learn a decoder D_θ generating an image, *i.e.*, $\tilde{\mathbf{x}} = D_\theta(\mathbf{z})$, and this image is re-encoded in a hidden representation: $\mathbf{h} = R(\tilde{\mathbf{x}})$, and compared to the original image in the same space using a normal distribution. Once trained, the weights of the encoder-decoder are fixed for the rest of the steps of our algorithm.

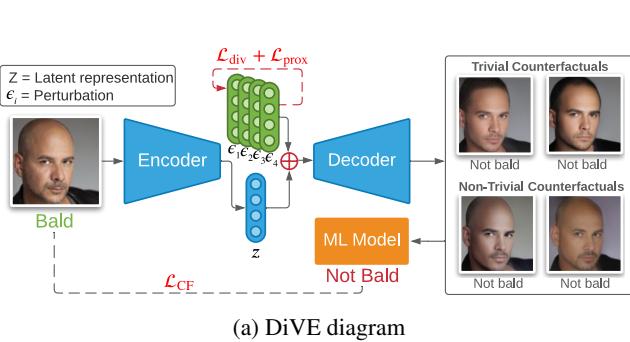
3.2. Interpreting the ML model

In order to find weaknesses in the ML model, DiVE searches for a collection of n latent perturbations $\{\epsilon_i\}_{i=1}^n$ such that the decoded output $\tilde{\mathbf{x}}_i = D_\theta(\mathbf{z} + \epsilon_i)$ yields a specific response from the ML model, *i.e.*, $f(\tilde{\mathbf{x}}_i) = \tilde{y}_i$ for any chosen $\tilde{y}_i \in [0, 1]$. We optimize ϵ_i 's by minimizing:

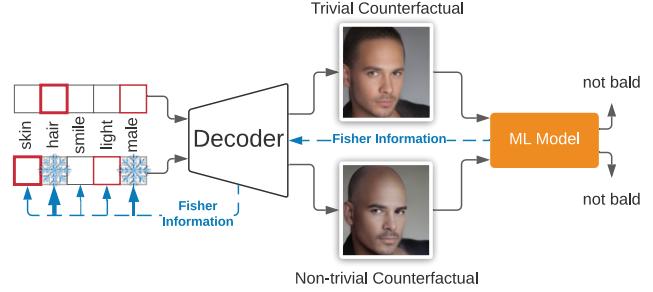
$$\begin{aligned} \mathcal{L}_{\text{DiVE}}(\mathbf{x}, \tilde{y}, \{\epsilon_i\}_{i=1}^n) &= \sum_i \mathcal{L}_{\text{CF}}(\mathbf{x}, \tilde{y}, \epsilon_i) \\ &\quad + \lambda \cdot \sum_i \mathcal{L}_{\text{prox}}(\mathbf{x}, \epsilon_i) \\ &\quad + \alpha \cdot \mathcal{L}_{\text{div}}(\{\epsilon_i\}_{i=1}^n), \end{aligned} \quad (1)$$

where λ , and α determine the relative importance of the losses. Minimization is performed with gradient descent and the complete algorithm can be found in the Supplementary Material. We now describe the different loss terms.

Counterfactual loss. The goal of this loss function is to identify a change of latent attributes that will cause the ML model f to change its prediction. For example, in face recognition, if the classifier detects that there is a smile present whenever the hair is brown, then this loss function



(a) DiVE diagram



(b) Effect of the Fisher Information

Figure 1: Left: DiVE encodes the input image to explain into a latent representation z . Then z is perturbed by ϵ and decoded as counterfactual examples. During training, \mathcal{L}_{CF} finds the set of ϵ that change the ML model classifier outcome while \mathcal{L}_{div} and $\mathcal{L}_{\text{prox}}$ enforce that the samples are *diverse* and *proximal*. These are four valid counterfactuals from the experiment in Section 4.1. However, only the bottom row contains counterfactuals where the man is still bald as indicated by the oracle or a human. These counterfactuals identify a weakness in the ML model. **Right:** Fisher Information indicates the most important latent directions for the ML model, where importance is represented by the thickness of the blue line (hair in this example). We keep those directions fixed since they are usually trivial and thus explanations modify other attributes (red boxes).

is likely to change the hair color attribute. This is achieved by sampling from the decoder $\tilde{\mathbf{x}} = D_{\theta}(\mathbf{z} + \epsilon)$, and optimizing the binary cross-entropy between the target \tilde{y} and the prediction $f(\tilde{\mathbf{x}})$:

$$\mathcal{L}_{\text{CF}}(\mathbf{x}, \tilde{y}, \epsilon) = \tilde{y} \cdot \log(f(\tilde{\mathbf{x}})) + (1 - \tilde{y}) \cdot \log(1 - f(\tilde{\mathbf{x}})). \quad (2)$$

Proximity loss. The goal of this loss function is to constrain the reconstruction produced by the decoder to be similar in appearance and attributes as the input. It consists of the following two terms,

$$\mathcal{L}_{\text{prox}}(\mathbf{x}, \epsilon) = \|\mathbf{x} - \tilde{\mathbf{x}}\|_1 + \gamma \cdot \|\epsilon\|_1, \quad (3)$$

where γ is a scalar weighting the relative importance of the two terms. The first term ensures that the explanations can be related to the input by constraining the input and the output to be similar. The second term aims to identify a sparse perturbation to the latent space \mathcal{Z} that confounds the ML model. This constrains the explainer to identify the least amount of attributes that affect the classifier’s decision in order to produce *sparse* explanations.

Diversity loss. This loss prevents the multiple explanations of the model from being identical. For instance, if gender and hair color are spuriously correlated with smile, the model should provide images either with different gender or different hair color. To achieve this, we jointly optimize for a collection of n perturbations $\{\epsilon_i\}_{i=1}^n$ and minimize their pairwise similarity:

$$\mathcal{L}_{\text{div}}(\{\epsilon_i\}_{i=1}^n) = \sqrt{\sum_{i \neq j} \left(\frac{\epsilon_i^T}{\|\epsilon_i\|_2} \frac{\epsilon_j}{\|\epsilon_j\|_2} \right)^2}. \quad (4)$$

The method resulting of optimizing Eq. 1 (DiVE) results in *diverse* counterfactuals that are more *valid*, *proximal*, and *sparse*. However, it may still produce *trivial* explanations, such as exaggerating a smile to explain a smile classifier without considering other valuable biases in the ML model such as hair color. While the diversity loss encourages the orthogonality of the explanations, there might still be several latent variables required to represent all variations of smile.

Beyond trivial counterfactual explanations. To find *non-trivial* explanations, we propose to prevent DiVE from perturbing the most influential latent factors of \mathcal{Z} on the ML model. We estimate the influence of each of the latent factors with the average Fisher information matrix:

$$\mathbf{F} = \mathbb{E}_{p(i)} \mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x}_i)} \mathbb{E}_{p(y|\mathbf{z})} \nabla_{\mathbf{z}} \ln p(y|\mathbf{z}) \nabla_{\mathbf{z}} \ln p(y|\mathbf{z})^T, \quad (5)$$

where $p(y = 1|\mathbf{z}) = f(D_{\theta}(\mathbf{z}))$, and $p(y = 0|\mathbf{z}) = 1 - f(D_{\theta}(\mathbf{z}))$. The diagonal values of \mathbf{F} express the relative influence of each of the latent dimensions on the classifier output. Since the most influential dimensions are likely to be related to the main attribute used by the classifier, we propose to prevent Eq. 1 from perturbing them in order to find more surprising explanations. Thus when producing n explanations, we sort \mathcal{Z} by the magnitude of the diagonal, we partition it into n contiguous chunks that will be optimized for each of the explanations. We call this method DiVE_{Fisher}.

However, DiVE_{Fisher} does not guarantee that the different partitions of \mathcal{Z} all the factors concerning a *trivial* attribute are grouped together. Thus, we propose to partition \mathcal{Z} into subsets of latent factors that interact with each other when changing the predictions of the ML model. Such

interaction can be estimated using \mathbf{F} as an affinity measure. We use spectral clustering [59] to obtain a partition of \mathcal{Z} . This partition is represented as a collection of masks $\{\mathbf{m}_i\}_{i=1}^n$, where $\mathbf{m}_i \in \{0, 1\}^d$ represents which dimensions of \mathcal{Z} are part of cluster i . Finally, these masks are used in Equation 1 to bound each ϵ_i to its subspace, *i.e.*, $\epsilon'_i = \epsilon_i \circ \mathbf{m}_i$, where \circ represents element wise multiplication. Since these masks are orthogonal, this effectively replaces \mathcal{L}_{div} . In Section 4, we highlight the benefits of this clustering approach by comparing to other baselines. We call this method DiVE_{FisherSpectral}.

4. Experimental Results

In this section, we first evaluate the described methods on their ability to identify diverse *non-trivial* explanations for image misclassifications made by the ML model (Section 4.1) and the out-of-distribution performance of DiVE (Section 4.1). In the following sections we validate the correctness of DiVE by evaluating its performance on 4 different aspects: (1) the *validity* of the generated explanations as well as the ability to discover biases within the ML model and the data (Section 4.2); (2) their *proximity* in terms of FID, latent space closeness, and face verification accuracy (Section 4.3); and (3) the *sparsity* of the generated counterfactuals (Section 4.4).

Experimental Setup. To align with [9, 26, 57], we perform experiments on the CelebA database [36]. CelebA is a large-scale dataset containing more than 200K celebrity facial images. Each image is annotated with 40 binary attributes such as “Smiling”, “Male”, and “Eyeglasses”. These attributes allow us to evaluate counterfactual explanations by determining whether they could highlight spurious correlations between multiple attributes such as “lipstick” and “smile”. In this setup, explainability methods are trained in the training set and ML models are explained on the validation set. The hyperparameters of the explainer are searched by cross-validation on the training set. We compare our method with xGEM [26] and PE [57] as representatives of methods that use an unconditional generative model and a conditional GAN respectively. We use the same train and validation splits as PE [57]. DiVE and xGEM do not have access to the labeled attributes during training.

We test the out-of-distribution (OOD) performance of DiVE with the Symbols dataset [32]. Symbols is an image generator with characters from the Unicode standard and the wide range of artistic fonts provided by the open font community. This grants us better control the features present in each set when compared to CelebA. We generate 100K black and white of 32×32 images from 48 characters in the latin alphabet and more than 1K fonts. We use the character type to create disjoint sets for OOD training and we use the fonts to introduce biases in the data. We provide

a sample of the dataset in the Supplementary Material.

We compare three version of our method and two ablated version to three existing methods. DiVE, resulting of optimizing Eq. 1. DiVE_{Fisher}, which extends DiVE by using the Fisher information matrix introduced in Eq. 5. DiVE_{FisherSpectral}, which extends DiVE_{Fisher} with spectral clustering. We introduce two additional ablations of our method, DiVE-- and DiVE_{Random}. DiVE-- is equivalent to DiVE but using a pixel-based reconstruction loss instead of the perceptual loss. DiVE_{Random} uses random masks instead of using the Fisher information. Finally, we compare our baselines with xGEM as described in Joshi et al. [26], xGEM+, which is the same as xGem but uses the same auto-encoding architecture as DiVE, and PE as described by Singla et al. [57]. For our methods, we provide implementation details, architecture description, and algorithm in the Supplementary Material.

4.1. Beyond trivial explanations

Previous works on counterfactual generations tend to produce *trivial* input perturbations to change the output of the ML model. That is, they tend to increase/decrease the presence of the attribute that is intended to be classified. For instance, in Figure 3 all the explainers put a smile on the input face in order to increase the probability for “smile”. While that is correct, this explanation does not provide much insight about the potential weaknesses of the ML model. Instead, in this work we emphasize producing non-trivial explanations that are different from the main attribute that the ML model has been trained to identify. These kind of explanations provide more insight about the factors that affect the classifier and thus provide cues on how to improve the model or how to fix incorrect predictions.

To evaluate this, we propose a new benchmark that measures a method’s ability to generate *valuable* explanations. For an explanation to be valuable, it should 1) be misclassified by the ML model (*valid*), 2) not modify attributes intended to be classified by the ML model (*non-trivial*), and 3) not have diverged too much from the original sample (*proximal*). A misclassification provides insights into the weaknesses of the model. However, the counterfactual is even more insightful when it stays close to the original image as it singles-out spurious correlations learned by the ML model. Because it is costly to provide human evaluation of an automatic benchmark, we approximate both the proximity and the real class with the VGGFace2-based oracle. We choose the VGGFace2 model as it is less likely to share the same biases as the ML model, since it was trained for a different task than the ML model with an order of magnitude more data. We conduct a human evaluation experiment in the Supplementary Material, and we find a significant correlation between the oracle and the human predictions. For 1) and 2) we deem that an explanation is successful if the ML

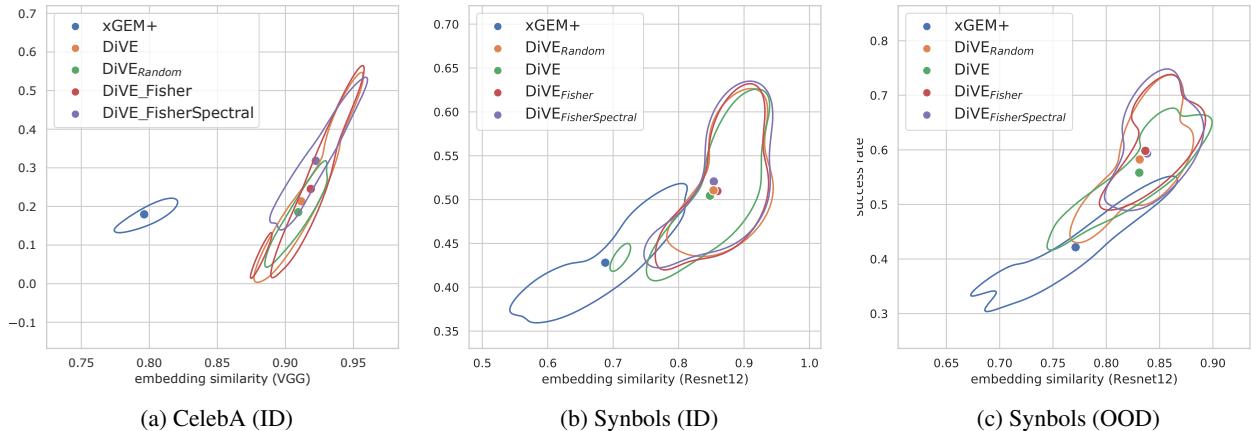


Figure 2: Beyond trivial explanations. Rate of successful explanations (y-axis) against embedding similarity (x-axis) for all methods. The most valuable explanations are in the top-right corner. We ran an hyperparameter sweep and denote the mean of the performances with a dot. The curves are computed with KDE. The left plot shows the performance on CelebA and the other two plots shows the performance for in-distribution (ID) and out-of-distribution (OOD) experiments on Symbols. All DiVE methods outperform xGEM+ on both metrics simultaneously when conditioning on *successful counterfactuals*. In both experiments, DiVE_{Fisher} and DiVE_{FisherSpectral} improve the performance over both DiVE_{Random} and DiVE.

model and the oracle make different predictions about the counterfactual. *E.g.*, the top counterfactuals in Figure 1a are not deemed successful explanations because both the ML model and the oracle agree on its class, however the two in the bottom row are successful because only the oracle made the correct prediction. These explanations where generated by DiVE_{FisherSpectral}. As for 3) we measure the proximity with the cosine distance between the sample and the counterfactual in the feature space of the oracle.

We test all methods from Section 4 on a subset of the CelebA validation set described in the Supplementary Material. We report the results of the full hyperparameter search. The vertical axis shows the success rate of the explainers, *i.e.*, the ratio of valid explanations that are non-trivial. This is the misclassification rate of the ML model on the explanations. The dots denote the mean performances and the curves are computed with Kernel Density Estimation (KDE). On average, DiVE improves the similarity metric over xGEM+ highlighting the importance of disentangled representations for identity preservation. Moreover, using information from the diagonal of the Fisher Information Matrix as described in Eq. 5 further improves the explanations as shown by the higher success rate of DiVE_{Fisher} over DiVE and DiVE_{Random}. Thus, preventing the model from perturbing the most influential latent factors helps to uncover spurious correlations that affect the ML model. Finally, the proposed spectral clustering of the full Fisher Matrix attains the best performance validating that the latent space partition can guide the gradient-based search towards better explanations. We reach the same conclusions in Table 3, where we provide a comparison with

PE for the attribute “Young”. In addition, we provide results for a version of xGEM+ with more disentangled latent factors (xGEM++). We find that disentangled representations provide the explainer with a more precise control on the semantic concepts being perturbed, which increases the success rate of the explainer by 16%.

Out-of-distribution generalization. In the previous experiments, the generative model of DiVE was trained on the same data distribution (*i.e.*, CelebA faces) as the ML model. We test the out-of-distribution performance of DiVE by training its auto-encoder on a subset of the latin alphabet of the Symbols dataset [32]. Then, counterfactual explanations are produced for a different disjoint subset of the alphabet. To evaluate the effectiveness of DiVE in finding biases on the ML model, we introduce spurious correlations in the data. Concretely, we assign a different set of fonts to each of the letters in the alphabet as detailed in the Supplementary Material. In-distribution (ID) results are reported in Figure 2b for reference, and OD results are reported in Figure 2c. We observe that DiVE is able to find valuable counterfactuals even when the VAE was not trained on the same data distribution. Moreover, results are consistent with the CelebA experiment, with DiVE outperforming xGEM+ and Fisher information-based methods outperforming the rest.

4.2. Validity and bias detection

We evaluate DiVE’s ability to detect biases in the data. We follow the same procedure as PE [57] and train two binary classifiers for the attribute “Smiling”. The first one is trained on a biased version of CelebA where males are smil-

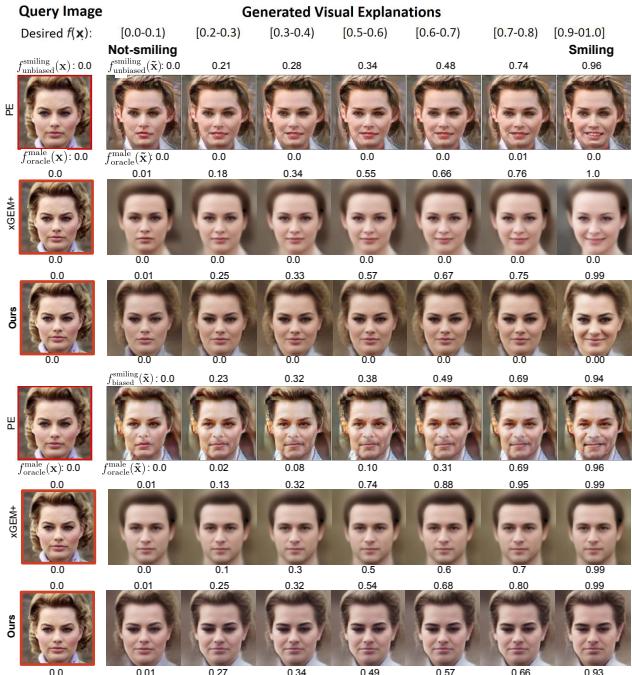


Figure 3: Bias detection experiment. Columns present explanations for a target “Smiling” probability interval. Rows contain explanations produced by PE [57], xGEM+ and our DiVE. (a) of a gender-unbiased classifier, and (b) a gender-biased “Smile” classifier. The classifier output probability is displayed on top of the images while the oracle prediction for gender is displayed at the bottom.

ing and females are not smiling (f_{biased}). This reflects an existing bias in the data gathering process where female are usually expected to smile [9, 19]. The second one is trained on the unbiased version of the data ($f_{unbiased}$). Both classifiers are evaluated on the CelebA validation set. Also following Singla et al. [57], we train an oracle classifier (f_{oracle}) based on VGGFace2 [3] which obtains perfect accuracy on the gender attribute. The hypothesis is that if “Smiling” and “Gender” are confounded by the classifier, so should be the explanations. Therefore, we could identify biases when the generated examples not only change the target attribute but also the confounded one. To generate counterfactuals, DiVE produces perturbations until it changes the original prediction of the classifier (“Smiling” to “Non-Smiling”). As described by Singla et al. [57] only *valid* explanations are considered, i.e. those that change the original prediction of the classifier.

We follow the procedure introduced in [26, 57] and report a confounding metric for bias detection in Table 1. The columns *Smiling* and *Non-Smiling* indicate the target class for counterfactual generation. The rows *Male* and *Female* contain the proportion of counterfactuals that are classified by the oracle as “Male” and “Female”. We can see that the generated explanations for f_{biased} are classified more of-

ten as “Male” when the target attribute is “Smiling”, and “Female” when the target attribute is “Non-Smiling”. The confounding metric, denoted as *overall*, is the fraction of generated explanations for which the gender was changed with respect to the original image. It thus reflect the magnitude of the the bias as approximated by the explainers.

Singla et al. [57] consider that a model is better than another if the confounding metric is the highest on f_{biased} and the lowest on $f_{unbiased}$. However, they assume that f_{biased} always predicts the “Gender” based on “Smile”. Instead, we propose to evaluate the confounding metric by comparing it to the empirical bias of the model, denoted as ground truth in Table 1. Details provided in the Supplementary Material.

We observe that DiVE is more successful than PE at detecting biases although the generative model of DiVE was not trained with the biased data. While xGEM+ has a higher success rate at detecting biases in some cases, it produces lower-quality images that are far from the input. In Figure 3, we provide samples generated by our method with the two classifiers and compare them to PE and xGEM+. We found that gender changes with the “Smiling” attribute with f_{biased} while for $f_{unbiased}$ it stayed the same. In addition, we also observed that for f_{biased} the correlation between “Smile” and “Gender” is higher than for PE. It can also be observed that xGEM+ fails to retain the identity of the person in \mathbf{x} when compared to PE and our method. Qualitative results are reported in Figure 3.

4.3. Counterfactual Explanation Proximity

We evaluate the *proximity* of the counterfactual explanations using FID scores [20] on CelebA as described by Singla et al. [57] (we observed similar results on MNIST and CIFAR [31, 33]). The scores are based on the target attributes “Smiling” and “Young”, and are divided into 3 categories: *Present*, *Absent*, and *Overall*. *Present* considers explanations for which the ML model outputs a probability greater than 0.9 for the target attribute. *Absent* refers to explanations with a probability lower than 0.1. *Overall* considers all the successful counterfactuals, which changed the original prediction of the ML model.

We report these scores in Table 2 for all 3 categories. DiVE produces the best quality counterfactuals, surpassing PE by 6.3 FID points for the “Smiling” target and 19.6 FID points for the “Young” target in the *Overall* category. DiVE obtains lower FID than xGEM+ which shows that the improvement not only comes from the superior architecture of our method. Further, there are two other factors that explain the improvement of DiVE’s FID. First, the β -TCVAE decomposition of the KL divergence improves the disentanglement ability of the model while suffering less reconstruction degradation than the VAE. Second, the perceptual loss makes the image quality constructed by DiVE to be comparable with that of the GAN used in PE. Additional exper-

Table 1: Bias detection experiment. Ratio of generated counterfactuals classified as “Smiling” and “Non-Smiling” for a classifier biased on gender (f_{biased}) and an unbiased classifier (f_{unbiased}). Bold indicates *Overall* closest to *Ground truth* (detailed in the Appendix).

ML model	Target label					
	Smiling			Non-Smiling		
	PE	xGEM+	DiVE	PE	xGEM+	DiVE
f_{biased}	Male	0.52	0.94	0.89	0.18	0.24
	Female	0.48	0.06	0.11	0.82	0.77
	Overall	0.12	0.29	0.22	0.35	0.33
Ground truth		0.75		0.67		
f_{unbiased}	Male	0.48	0.41	0.42	0.47	0.38
	Female	0.52	0.59	0.58	0.53	0.62
	Overall	0.07	0.13	0.10	0.08	0.15
Ground truth		0.04		0.00		

Table 2: FID of DiVE compared to xGEM [26], Progressive Exaggeration (PE) [57], xGEM trained with our backbone (xGEM+), and DiVE trained without the perceptual loss (DiVE--)

Target Attribute	xGEM	PE	xGEM+	DiVE--	DiVE
Smiling					
Present	111.0	46.9	67.2	54.9	30.6
Absent	112.9	56.3	77.8	62.3	33.6
Overall	106.3	35.8	66.9	55.9	29.4
Young					
Present	115.2	67.6	68.3	57.2	31.8
Absent	170.3	74.4	76.1	51.1	45.7
Overall	117.9	53.4	59.5	47.7	33.8

iments in the Supplementary Material show that DiVE is more successful at preserving the identity of the faces than PE and xGEM and thus at producing feasible explanations. These results suggest that the combination of disentangled latent features and the regularization of the latent features help DiVE to produce the minimal perturbations of the input that produce a successful counterfactual.

In Figure 3 we show qualitative results obtained by targeting different probability ranges for the output of the ML model as described in PE. DiVE produces more natural-looking facial expressions than xGEM+ and PE. Additional results for “Smiling” and “Young” are provided in the Supplementary Material.

4.4. Counterfactual Explanation Sparsity

We quantitatively compare the amount of valid and sparse counterfactuals provided by different baselines. Table 3 shows the results for a classifier model trained on the attribute “Young” of the CelebA dataset. The first row shows the number of attributes that each method change in average to generate a valid counterfactual. At-

Table 3: Average number of attributes changed per explanation and percentage of non-trivial explanations. This experiment evaluates the counterfactuals generated by different methods for an ML model trained on the attribute “Young” of the CelebA dataset. xGEM++ is xGEM+ using β -TCVAE as generator.

	PE [57]	xGEM+ [26]	xGEM++	DiVE	DiVE _{Fisher}	DiVE _{FisherSpectral}
Attr. change	03.74	06.92	06.70	04.81	04.82	04.58
Non-trivial (%)	05.12	18.56	34.62	43.51	42.99	51.07

tribute changes is measured from the output of the with the VGGFace2-based oracle. Methods that require to change less attributes are likely to be actionable by a user. We observe that DiVE changes less attributes on average than xGEM+. DiVE_{FisherSpectral} is the method that changes less attributes. To better understand the effect of disentangled representations, we also report results for a version of xGEM+ with the β -TCVAE backbone (xGEM++). We do not observe significant effects on the sparsity of the counterfactuals. In fact, a fine-grained decomposition of concepts in the latent space could lead to lower the sparsity.

5. Limitations and Future Work

This work shows that a good generative model can provide interesting insights on the biases of an ML model. However, this relies on a properly disentangled representation. In the case where the generative model is heavily entangled it would fail to produce explanations with a sparse amount of features. However, our approach can still tolerate a small amount of entanglement, yielding a small decrease in interpretability. We expect that progress in identifiability [28, 38] will increase the quality of representations. With a perfectly disentangled model, our approach could still miss some explanations or biases. *E.g.*, with the spectral clustering of the Fisher, we group latent variables and only produce a single explanation per group in order to present explanations that are conceptually different. This may leave behind some important explanations, but the user can simply increase the number of clusters or the number of explanations per clusters for a more in-depth analysis.

In addition, finding the optimal hyperparameters for the VAE and their OOD generalization is an open problem. If the generative model is trained on biased data, one could expect the counterfactuals to be biased as well. However, as shown in Figure 2c, our model still finds non-trivial explanations when applied OOD.

Although the generative model plays an important role to produce valuable counterfactuals in the image domain, our work could be extended to other domains. For example, Eq. 1 could be applied on tabular data by directly optimizing observed features instead of latent factors of a VAE. However, further work would be needed to adapt DiVE to produce perturbations on discrete and categorical variables.

References

- [1] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, 2018. 2, 11
- [2] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018. 13
- [3] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *International Conference on Automatic Face and Gesture Recognition*, 2018. 7, 13
- [4] C.-H. Chang, E. Creager, A. Goldenberg, and D. Duvenaud. Explaining image classifiers by counterfactual generation. In *International Conference on Learning Representations*, 2019. 1, 2, 3, 11
- [5] R. T. Chen, X. Li, R. B. Grosse, and D. K. Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, 2018. 2, 3, 11
- [6] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995. 2
- [7] P. Dabkowski and Y. Gal. Real time image saliency for black box classifiers. *arXiv preprint arXiv:1705.07857*, 2017. 2, 3, 11
- [8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition*, 2009. 13
- [9] E. Denton, B. Hutchinson, M. Mitchell, and T. Gebru. Detecting bias with generative counterfactual face attribute augmentation. *arXiv preprint arXiv:1906.06439*, 2019. 2, 5, 7, 11
- [10] A. Dhurandhar, P.-Y. Chen, R. Luss, C.-C. Tu, P. Ting, K. Shanmugam, and P. Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In *Advances in Neural Information Processing Systems*, pages 592–603, 2018. 12
- [11] R. C. Fong and A. Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *International Conference on Computer Vision*, 2017. 1, 2, 11
- [12] H. Fu, C. Li, X. Liu, J. Gao, A. Celikyilmaz, and L. Carin. Cyclical annealing schedule: A simple approach to mitigating kl vanishing. *arXiv preprint arXiv:1903.10145*, 2019. 13
- [13] Y. Gal, J. Hron, and A. Kendall. Concrete dropout. In *Advances in neural information processing systems*, 2017. 1
- [14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014. 2, 11
- [15] Y. Goyal, Z. Wu, J. Ernst, D. Batra, D. Parikh, and S. Lee. Counterfactual visual explanations. *arXiv preprint arXiv:1904.07451*, 2019. 2, 11
- [16] R. Guidotti, A. Monreale, F. Giannotti, D. Pedreschi, S. Ruggieri, and F. Turini. Factual and counterfactual explanations for black box decision making. *IEEE Intelligent Systems*, 34(6):14–23, 2019. 11
- [17] R. Guidotti, A. Monreale, S. Matwin, and D. Pedreschi. Black box explanation by learning image exemplars in the latent feature space. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 189–205. Springer, 2019. 11
- [18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, 2016. 2
- [19] D. Hestenes and I. Halloun. Interpreting the force concept inventory: A response to march 1995 critique by huffman and heller. *The physics teacher*, 33(8):502–502, 1995. 7
- [20] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, 2017. 2, 7
- [21] X. Hou, L. Shen, K. Sun, and G. Qiu. Deep feature consistent variational autoencoder. In *Winter Conference on Applications of Computer Vision*, 2017. 3
- [22] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Computer Vision and Pattern recognition*, 2017. 13
- [23] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 2015. 13
- [24] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In *Computer Vision and Pattern Recognition Workshops*, 2017. 2
- [25] F. V. Jensen et al. *An introduction to Bayesian networks*, volume 210. UCL press London, 1996. 2
- [26] S. Joshi, O. Koyejo, B. Kim, and J. Ghosh. xgems: Generating exemplars to explain black-box models. *arXiv preprint arXiv:1806.08867*, 2018. 1, 2, 5, 7, 8, 11, 12, 14, 15
- [27] A.-H. Karimi, G. Barthe, B. Balle, and I. Valera. Model-agnostic counterfactual explanations for consequential decisions. In *International Conference on Artificial Intelligence and Statistics*, pages 895–905, 2020. 11
- [28] I. Khemakhem, D. Kingma, R. Monti, and A. Hyvärinen. Variational autoencoders and nonlinear ica: A unifying framework. In *International Conference on Artificial Intelligence and Statistics*, pages 2207–2217. PMLR, 2020. 8
- [29] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 13
- [30] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2, 3
- [31] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009. 7
- [32] A. Lacoste, P. Rodríguez López, F. Branchaud-Charron, P. Atighehchian, M. Caccia, I. H. Laradji, A. Drouin, M. Craddock, L. Charlin, and D. Vázquez. Symbols: Probing learning algorithms with synthetic datasets. *Advances in Neural Information Processing Systems*, 33, 2020. 5, 6, 16
- [33] Y. LECUN. The mnist database of handwritten digits. <http://yann.lecun.com/exdb/mnist/>. URL <https://ci.nii.ac.jp/naid/10027939599/en/>. 7
- [34] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989. 2

- [35] S. Liu, B. Kailkhura, D. Loveland, and Y. 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
- [36] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *International Conference on Computer Vision*, 2015. 2, 5
- [37] F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Schölkopf, and O. Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*, 2019. 3
- [38] F. Locatello, B. Poole, G. Rätsch, B. Schölkopf, O. Bachem, and M. Tschannen. Weakly-supervised disentanglement without compromises. *arXiv preprint arXiv:2002.02886*, 2020. 8
- [39] J. Lucas, G. Tucker, R. Grosse, and M. Norouzi. Understanding posterior collapse in generative latent variable models. 2019. 13
- [40] S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, 2017. 1, 2, 11
- [41] R. K. Mothilal, A. Sharma, and C. Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Conference on Fairness, Accountability, and Transparency*, 2020. 1, 3, 12
- [42] J. A. Nelder and R. W. Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, 135(3):370–384, 1972. 2
- [43] B. Oreshkin, P. Rodríguez López, and A. Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. *Advances in Neural Information Processing Systems*, 31:721–731, 2018. 15
- [44] N. Papernot and P. McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*, 2018. 2
- [45] M. Pawelczyk, K. Broelemann, and G. Kasneci. Learning model-agnostic counterfactual explanations for tabular data. In *Proceedings of The Web Conference 2020*, pages 3126–3132, 2020. 2, 12
- [46] N. Pawłowski, D. C. Castro, and B. Glocker. Deep structural causal models for tractable counterfactual inference. *arXiv preprint arXiv:2006.06485*, 2020. 3
- [47] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. Film: Visual reasoning with a general conditioning layer. *arXiv preprint arXiv:1709.07871*, 2017. 13
- [48] R. Poyiadzi, K. Sokol, R. Santos-Rodriguez, T. De Bie, and P. Flach. Face: feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 344–350, 2020. 12
- [49] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986. 2
- [50] P. Ramachandran, B. Zoph, and Q. V. Le. Searching for activation functions. *International Conference on Learning Representations*, 2018. 13
- [51] M. T. Ribeiro, S. Singh, and C. Guestrin. “why should i trust you?” explaining the predictions of any classifier. In *International Conference on Knowledge Discovery and Data Mining*, 2016. 1, 2, 11
- [52] C. Russell. Efficient search for diverse coherent explanations. In *Conference on Fairness, Accountability, and Transparency*, 2019. 1
- [53] A. Sauer and A. Geiger. Counterfactual generative networks. *arXiv preprint arXiv:2101.06046*, 2021. 2
- [54] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *International Conference on Computer Vision*, 2017. 1, 2, 11
- [55] A. Shrikumar, P. Greenside, and A. Kundaje. Learning important features through propagating activation differences. *arXiv preprint arXiv:1704.02685*, 2017. 1, 2, 11
- [56] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013. 2, 11
- [57] S. Singla, B. Pollack, J. Chen, and K. Batmanghelich. Explanation by progressive exaggeration. In *International Conference on Learning Representations*, 2020. 1, 2, 3, 5, 6, 7, 8, 12, 13, 14, 15
- [58] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014. 2, 11
- [59] X. Y. Stella and J. Shi. Multiclass spectral clustering. In *null*, page 313. IEEE, 2003. 5
- [60] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 13
- [61] A. Van Looveren and J. Klaise. Interpretable counterfactual explanations guided by prototypes. *arXiv preprint arXiv:1907.02584*, 2019. 12
- [62] A. Van Looveren, J. Klaise, G. Vacanti, and O. Cobb. Conditional generative models for counterfactual explanations. *arXiv preprint arXiv:2101.10123*, 2021. 2
- [63] F. Yang, N. Liu, M. Du, and X. Hu. Generative counterfactuals for neural networks via attribute-informed perturbation. *arXiv preprint arXiv:2101.06930*, 2021. 2
- [64] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014. 2, 11
- [65] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Computer Vision and Pattern Recognition*, 2016. 2, 11

Supplementary Material

Section A contains the extended related work, Section B shows additional qualitative results, Section C contains additional results for identity preservation, Section D contains the implementation details, Section E contains additional information about the experimental setup, Section F provides the results of human evaluation of DiVE, Section G contains details about the model architecture, Section H contains the DiVE Algorithm, and Section J contains details about the OOD experiment.

A. Extended Related Work

Counterfactual explanation lies inside a more broadly-connected body of work for explaining classifier decisions. Different lines of work share this goal but vary in the assumptions they make about what elements of the model and data to emphasize as way of explanation.

Model-agnostic counterfactual explanation like [40, 51], these models make no assumptions about model structure, and interact solely with its label predictions. Karimi et al. [27] develop a model agnostic, as well as metric agnostic approach. They reduce the search for counterfactual explanations (along with user-provided constraints) into a series of satisfiability problems to be solved with off-the-shelf SAT solvers. Similar in spirit to [51], Guidotti et al. [16] first construct a local neighbourhood around test instances, finding both positive and negative exemplars within the neighbourhood. These are used to learn a shallow decision tree, and explanations are provided in terms of the inspection of its nodes and structure. Subsequent work builds on this local neighbourhood idea [17], but specializes to medical diagnostic images. They use a VAE to generate both positive and negative samples, then use random heuristic search to arrive at a balanced set. The generated explanatory samples are used to produce a saliency feature map for the test data point by considering the median absolute deviation of pixel-wise differences between the test point, and the positive and negative example sets.

Gradient based feature attribution. These methods identify input features responsible for the greatest change in the loss function, as measured by the magnitude of the gradient with respect to the inputs. Early work in this area focused on how methodological improvements for object detection in images could be re-purposed for feature attribution [64, 65], followed by work summarized gradient information in different ways [54, 56, 58]. Closer inspection identified pitfalls of gradient-based methods, including induced bias due to gradient saturation or network structure [1], as well as discontinuity due to activation functions [55]. These methods typically produce dense fea-

ture maps, which are difficult to interpret. In our work we address this by constraining the generative process of our counterfactual explanations.

Reference based feature attribution. These methods focus instead on measuring the differences observed by substituting observed input values with ones drawn from some reference distribution, and accumulating the effects of these changes as they are back-propagated to the input features. Shrikumar et al. [55] use a modified back-propagation approach to gracefully handle zero gradients and negative contributions, but leave the reference to be specified by the user. Fong and Vedaldi [11] propose three different heuristics for reference values: replacement with a constant, addition of noise, and blurring. Other recent efforts have focused on more complex proposals of the reference distribution. Chen et al. [5] construct a probabilistic model that acts as a lower bound on the mutual information between inputs and the predicted class, and choose zero values for regions deemed uninformative. Building on desiderata proposed by Dabkowski and Gal [7], Chang et al. [4] use a generative model to marginalize over latent values of relevant regions, drawing plausible values for each. These methods typically either do not identify changes that would *alter* a classifier decision, or they do not consider the plausibility of those changes.

Counterfactual explanations. Rather than identify a set of features, counterfactual explanation methods instead generate perturbed versions of observed data that result in a corresponding change in model prediction. These methods usually assume both more access to model output and parameters, as well as constructing a generative model of the data to find trajectories of variation that elucidate model behaviour for a given test instance.

Joshi et al. [26] propose a gradient guided search in latent space (via a learned encoder model), where they progressively take gradient steps with respect to a regularized loss that combines a term for plausibility of the generated data, and the loss of the ML model. Denton et al. [9] use a Generative Adversarial Network (GAN) [14] for detecting bias present in multi-label datasets. They modify the generator to obtain latent codes for different data points and learn a linear decision boundary in the latent space for each class attribute. By sampling generated data points along the vector orthogonal to the decision boundary, they observe how crossing the boundary for one attribute causes undesired changes in others. Some counterfactual estimation methods forego a generative model by instead solving a surrogate editing problem. Given an original image (with some predicted class), and an image with a desired class prediction value, Goyal et al. [15] produce a counterfactual explanation through a series of edits to the original image by value

substitutions in the learned representations of both images. Similar in spirit are Dhurandhar et al. [10] and Van Looveren and Klaise [61]. The former propose a search over features to highlight subsets of those present in each test data point that are typically present in the assigned class, as well as features usually absent in examples from adjacent classes (instances of which are easily confused with the label for the test point predicted by the model). The latter generate counterfactual data that is proximal to x_{test} , with a sparse set of changes, and close to the training distribution. Their innovation is to use class prototypes to serve as an additional regularization term in the optimization problem whose solution produces a counterfactual.

Several methods go beyond providing counterfactually generated data for explaining model decisions, by additionally qualifying the effect of proposed changes between a test data point and each counterfactual. Mothilal et al. [41] focus on tabular data, and generate sets of counterfactual explanations through iterative gradient based improvement, measuring the cost of each counterfactual by either distance in feature space, or the sparsity of the set of changes (while also allowing domain expertise to be applied). Poyiadzi et al. [48] construct a weighted graph between each pair of data point, and identify counterfactuals (within the training data) by finding the shortest paths from a test data point to data points with opposing classes. Pawelczyk et al. [45] focus on modelling the density of the data to provide ‘attainable’ counterfactuals, defined to be proximal to test data points, yet not lying in low-density sub-spaces of the data. They further propose to weigh each counterfactual by the changes in percentiles of the cumulative distribution function for each feature, relative to the value of a test data point.

B. Qualitative results

Figure 4,5 present counterfactual explanations for additional persons and attributes. The results show that DiVE achieves higher quality reconstructions compared to other methods. Further, the reconstructions made by DiVE are more correlated with the desired target for the ML model output $f(x)$. We compare DiVE to PE and xGEM+. We found that gender changes with the “Smiling” attribute with f_{biased} while for $f_{unbiased}$ it stayed the same. In addition, we also observed that for f_{biased} the correlation between “Smile” and “Gender” is higher than for PE. It can also be observed that xGEM+ fails to retain the identity of the person in x when compared to PE and our method. Finally, Figure 6 shows *successful counterfactuals* for different instantiations of DiVE.

Note that PE directly optimizes the generative model to take an input variable $\delta \in \mathbb{R}$ that defines the desired output probability $\tilde{y} = f(\mathbf{x}) + \delta$. To obtain explanations at different probability targets, we train a second order spline on the trajectory of perturbations produced during the gradient

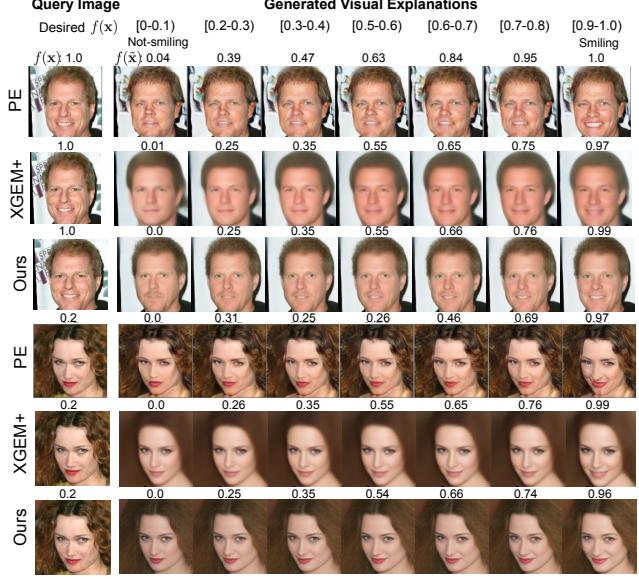


Figure 4: Qualitative results of DiVE, Progressive Exaggeration (PE) [57], and xGEM [26] for the “Smiling” attribute. Each column shows the explanations generated for a target probability output of the ML model. The numbers on top of each row show the actual output of the ML model.

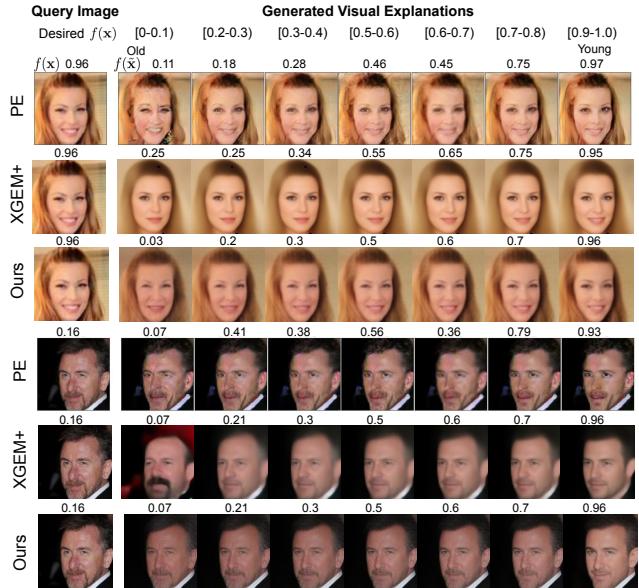


Figure 5: Qualitative results of DiVE, Progressive Exaggeration (PE) [57], and xGEM+ for the “Young” attribute. Each column shows the explanations generated for a target probability output of the ML model. The numbers on top of each row show the actual output of the ML model.

descent steps of our method. Thus, given the set of perturbations $\{\epsilon_t\}$, $\forall t \in 1..T$, obtained during T gradient steps, and the corresponding black-box outputs $\{f(y|\epsilon_t)\}$, the spline

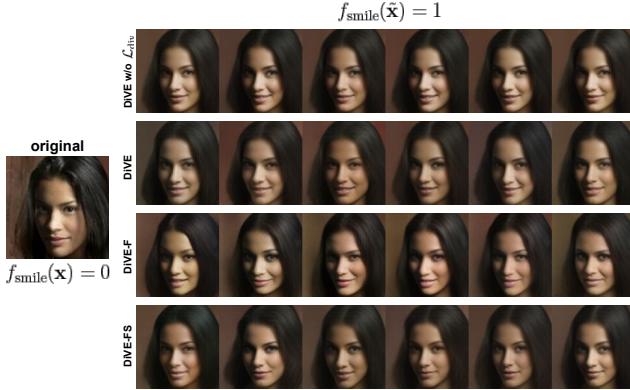


Figure 6: Successful counterfactual generations for different instantiations of DiVE. Here, the original image was misclassified as non-smiling. All methodologies were able to change the predicted class to "Smiling".

obtains the $\varepsilon_{\tilde{y}}$ for a target output \tilde{y} by interpolation.

C. Identity preservation

As argued, valuable explanations should remain proximal to the original image. Accordingly, we performed the identity preservation experiment found in [57] to benchmark the methodologies against each other. Specifically, use the VGGFace2-based [3] oracle to extract latent codes for the original images as well as for the explanations and report **latent space closeness** as the fraction of time the explanations' latent codes are the closest to their respective original image latent codes' compared to the explanations on different original images. Further, we report **face verification** accuracy which consist of the fraction of time the cosine distance between the aforementioned latent codes is below 0.5.

Table 4 presents both metrics for DiVE and its baselines on the "Smiling" and "Young" classification tasks. We find that DiVE outperforms all other methods on the "Young" classification task and almost all on the "Smiling" task.

D. Implementation details

In this Section, we provide provide the details to ensure the that our method is reproducible.

Architecture details. DiVE's architecture is a variation BigGAN [2] as shown in Table 6. We chose this architecture because it achieved impressive FID results on the ImageNet [8]. The decoder (Table 6b) is a simplified version of the 128×128 BigGAN's residual generator, without non-local blocks nor feature concatenation. We use InstanceNorm [60] instead of BatchNorm [23] to obtain consistent outputs at inference time without the need of an additional mechanism such as recomputing statistics [2]. All the InstanceNorm operations of the decoder are conditioned on

the input code \mathbf{z} in the same way as FILM layers [47]. The encoder (Table 6a) follows the same structure as the BigGAN 128×128 discriminator with the same simplifications done to our generator. We use the Swish non-linearity [50] in all layers except for the output of the decoder, which uses a Tanh activation.

For all experiments we use a latent feature space of 128 dimensions. The ELBO has a natural principled way of selecting the dimensionality of the latent representation. If d is larger than necessary, it will not enhance the reconstruction error and the optimization of the ELBO will make the posterior equal to the prior for these extra dimensions. More can be found on the topic in [39]. In practice, we experimented with $d = \{64, 128, 256\}$ and found that with $d = 128$ we achieved a slightly lower ELBO.

To project the 2d features produced by the encoder to a flat vector $(\mu, \log(\sigma^2))$, and to project the sampled codes \mathbf{z} to a 2d space for the decoder, we use 3-layer MLPs. For the face attribute classifiers, we use the same DenseNet [22] architecture as described in Progressive Exaggeration [57].

Optimization details. All the models are optimized with Adam [29] with a batch size of 256. During the training step, the auto-encoders are optimized for 400 epochs with a learning rate of $4 \cdot 10^{-4}$. The classifiers are optimized for 100 epochs with a learning rate of 10^{-4} . To prevent the auto-encoders from suffering KL vanishing, we adopt the cyclical annealing schedule proposed by Fu et al. [12].

Counterfactual inference details. At inference time, the perturbations are optimized with Adam until the ML model output for the generated explanation $f(\tilde{\mathbf{x}})$ only differs from the target output \tilde{y} by a margin δ or when a maximum number of iterations τ is reached. We set $\tau = 20$ for all the experiments since more than 90% of the counterfactuals are found after that many iterations. The different ϵ_i are initialized by sampling from a normal distribution $\mathcal{N} \sim (0, 0.01)$. For the DiVE_{Fisher} baseline, to identify the most valuable explanations, we sort ϵ by the magnitude of the diagonal of the Fisher Information Matrix, i.e. $\mathbf{f} = \text{diag}(\mathbf{F})$. Then, we divide the dimensions of the sorted ϵ into N contiguous partitions of size $k = \frac{D}{N}$, where D is the dimensionality of \mathcal{Z} . Formally, let $\epsilon^{(f)}$ be ϵ sorted by \mathbf{f} , then $\epsilon^{(f)}$ is constrained as follows,

$$\varepsilon_{i,j}^{(f)} = \begin{cases} 0, & \text{if } j \in [(i-1) \cdot k, i \cdot k] \\ \varepsilon_{i,j}^{(f)}, & \text{otherwise} \end{cases}, \quad (6)$$

where $i \in 1..N$ indexes each of the multiple ϵ , and $j \in 1..D$ indexes the dimensions of ϵ . As a result we obtain partitions with different order of complexity. Masking the first partition results in explanations that are most implicit

	CelebA:Smiling				CelebA:Young			
	xGEM	PE	xGEM+	DiVE (ours)	xGEM	PE	xGEM+	DiVE (ours)
Latent Space Closeness	88.2	88.0	99.8	98.7	89.5	81.6	97.5	99.1
Face Verification Accuracy	0.0	85.3	91.2	97.3	0.0	72.2	97.4	98.2

Table 4: Identity preserving performance on two prediction tasks.

within the model and the data. On the other hand, masking the last partition results in explanations that are more explicit.

To compare with Singla et al. [57] in Figures 4-5 we produced counterfactuals at arbitrary target values \tilde{y} of the output of the ML model classifier. One way to achieve this would be to optimize \mathcal{L}_{CF} for each of the target probabilities. However, these successive optimizations would slow down the process of counterfactual generation. Instead, we propose to directly maximize the target class probability and then interpolate between the points obtained in the gradient descent trajectory to obtain the latent factors of the different target probabilities. Thus, given the set of perturbations $\{\epsilon_t\}$, $\forall t \in 1..T$, obtained during T gradient steps, and the corresponding ML model outputs $\{f(y|\epsilon_t)\}$, we obtain the $\epsilon_{\tilde{y}}$ for a target output \tilde{y} by interpolation. We do such interpolation by fitting a piecewise quadratic polynomial on the latent trajectory, commonly known as Spline in the computer graphics literature.

E. Beyond Trivial Explanations Experimental Setup

The experimental benchmark proposed in Section 4.1 is performed on a subset of the validation set of CelebA. This subset is composed of 4 images for each CelebA attribute. From these 4 images, 2 were correctly classified by the ML model, while the other 2 were misclassified. The two correctly classified images are chosen so that one was classified with a high confidence of 0.9 and the other one with low confidence of 0.1. The 2 misclassifications were chosen with the same criterion. The total size of the dataset is of 320 images. For each of these images we generate k counterfactual explanations. From these counterfactuals, we report the ratio of successful explanations.

Here are the specific values we tried in our hyperparameter search: $\gamma \in [0.0, 0.001, 0.1, 1.0]$, $\alpha \in [0.0, 0.001, 0.1, 1.0]$, $\lambda \in [0.0001, 0.0005, 0.001]$, number of explanations 2 to 15 and learning rate $\in [0.05, 0.1]$. Because xGEM+ does not have a γ nor α parameter, we increased its learning rate span to $[0.01, 0.05, 0.1]$ to reduce the gap in its search space compared with DiVE. We also changed the random seeds and ran a total of 256 trials.

Method	Human \neq ML Classifier (real non-trivial)	Correlation	p-value
xGEM+ [26]	38.37%	0.37	0.000
DiVE	38.65%	0.25	0.002
DiVE _{Random}	38.89%	0.24	0.001
DiVE _{Fisher}	40.56%	0.17	0.023
DiVE _{FisherSpectral}	41.90%	0.23	0.001

Table 5: Human evaluation. The first column contains the percentage of non-trivial counterfactuals from the perspective of the human oracle. These counterfactuals confuse the ML classifier without changing the main attribute being classified from the perspective of a human. The second column contains the Pearson correlation between the human and the oracle’s predictions. The third column contains the p-value for a t-test with the null hypothesis of the human and oracle predictions being uncorrelated.



Figure 7: Labelling interface. The user is presented with a counterfactual image and has to choose if the target attribute is present or not in the image.

F. Human Evaluation

We built a web-based human evaluation task to assess if DiVE is more successful at finding *non-trivial* counterfactuals than previous state of the art and the effectiveness of the VGG-based oracle, see Figure 7. For that, we present to a diverse set of 20 humans from different countries and backgrounds with valid counterfactuals and ask them whether the main attribute being classified by the ML model is present in the image or not. We use a subset of CelebA containing a random sample of 4 images per attribute, each one classified by the VGG_{Face} oracle as containing the attribute with the following levels of confidence: [0.1, 0.4, 0.6, 0.9]. From each of these 160 images, we generated counterfactuals with xGEM+ [26], DiVE, DiVE_{Random}, DiVE_{Fisher}, and

$\text{DiVE}_{\text{FisherSpectral}}$ and show the valid counterfactuals to the human annotators. Results are reported in Table 5. In the left column we observe that leveraging the Fisher information results in finding more non-trivial counterfactuals, which confuse the ML model without changing the main attribute being classified. In the second column we report the Pearson correlation between the oracle and the classifier predictions. A statistical inference test reveals a significant correlation ($p\text{-value} \leq 0.02$).

G. Model Architecture

Table 6 presents the architecture of the encoder and decoder used in DiVE.

H. Model Algorithm

Algorithm 1 presents the steps needed for DiVE to generate explanations for a given ML model using a sample input image.

DiVE’s objective is to discover biases on the ML model and the data. Thus, we use the *font* attribute in order to bias each of the characters on small disjoint subsets of fonts. Font subsets are chosen so that they are visually similar. In order to assess their similarity, we train a ResNet12 [43] to classify the fonts of the 100K images and calculate similarity in embedding space. Concretely, we use K-Means to obtain 16 clusters which are associated with each of the 16 characters used for counterfactual generation. The font assignments are reported in Table 7. Results for four different random counterfactuals are displayed in Figure 9. $\text{DiVE}_{\text{FisherSpectral}}$ successfully confuses the ML model without changing the oracle prediction, revealing biases of the ML model.

I. Details on the Bias Detection Metric

In Table 1 in the main text, we follow the procedure in first developed in [26] and adapted in [57] and report a confounding metric for bias detection. Namely, the “Male” and “Female” is the accuracy of the oracle on those class conditioned on the target label of the original image. For example, we can see that the generated explanations for the biased classifier, most methods generated an higher amount of Non-smiling females and smiling males, which was expected. The confounding metric, denoted as overall, is the fraction of generated explanations for which the gender was changed with respect to the original image. It thus reflect the magnitude of the the bias as approximated by the explainers. Singla et al. [57] consider that a model is better than another if the confounding metric is the highest on f_{biased} and the lowest on f_{unbiased} .

This is however not entirely true. There is no guarantee that f_{biased} will perfectly latch on the spurious correlation. In that case, an explainer’s ratio could potentially be

Table 6: DiCe architecture for 128×128 images. ch represents the channel width multiplier in each network.

$\text{RGB image } x \in \mathbb{R}^{128 \times 128 \times 3}$
ResBlock down $3ch \rightarrow 16ch$
ResBlock $16ch \rightarrow 32ch$
ResBlock down $32ch \rightarrow 32ch$
ResBlock $32ch \rightarrow 64ch$
ResBlock down $64ch \rightarrow 64ch$
ResBlock $64ch \rightarrow 128ch$
ResBlock down $128ch \rightarrow 128ch$
ResBlock $128ch \rightarrow 128ch$
ResBlock down $128ch \rightarrow 128ch$
IN, Swish, Linear $128ch \times 4 \times 4 \rightarrow 128ch$
IN, Swish, Linear $128ch \rightarrow 128ch$
IN, Swish, Linear $128ch \rightarrow 128ch \times 2$
$z \sim \mathcal{N}(\mu \in \mathbb{R}^{128}, \sigma \in \mathbb{R}^{128})$
(a) Encoder
$z \in \mathbb{R}^{128}$
Linear $128ch \rightarrow 128ch$
Linear $128ch \rightarrow 128ch$
Linear $128ch \rightarrow 128ch \times 4 \times 4$
ResBlock up $128ch \rightarrow 64ch$
ResBlock up $64ch \rightarrow 32ch$
ResBlock $32ch \rightarrow 16ch$
ResBlock up $16ch \rightarrow 16ch$
ResBlock $16ch \rightarrow 16ch$
ResBlock up $16ch \rightarrow 16ch$
ResBlock $16ch \rightarrow 16ch$
IN, Swish, Conv $16ch \rightarrow 3$ \tanh
(b) Decoder

too high which would reflect an overestimation of the bias. We thus need to a way to quantify the gender bias in each model. To do so, we look at the difference between the classifiers accuracy on “Smiling” when the image is of a “Male” versus a “Female”. Intuitively, the magnitude of this difference approximates how much the classifier latched on the “Male” attribute to make its smiling predictions. We compute the same metric for in the non-smiling case. We average both of them, which we refer as ground truth in Table 1 (main text). As expected, this value is high for the f_{biased} and

Algorithm 1: Generating Explanations

```

Input : Sample image  $x$ , ML model  $f(\cdot)$ 
Output : Generated Counterfactuals  $\tilde{x}$ 

1 Initialize the perturbations matrix parameter of size  $n \times d$ 
2  $\Sigma \leftarrow \text{randn}(\mu = 0, \sigma = 0.01)$ 

3 Get the original output from the ML model
4  $y \leftarrow f(x)$ 

5 Extract the latent features of the original input
6  $z \leftarrow q_\phi(x)$ 

7 Obtain fisher information on  $z$ 
8  $f_z \leftarrow F(z)$ 

9 Obtain  $k$  partitions using spectral clustering
10  $P \leftarrow \text{SpectralClustering}(f_z)$ 

11 Initialize counter
12  $i \leftarrow 0$ 

13 while  $i < \tau$  do
14   for each  $\varepsilon, p \in (\Sigma, P)$  do
15     Perturb the latent features
16      $\tilde{x} \leftarrow p_\theta(z + \varepsilon)$ 

17     Pass the perturbed image through the ML model
18      $\hat{y} \leftarrow f(\tilde{x})$ 

19     Learn to reconstruct  $\hat{Y}$  from  $Y$ 
20      $\mathcal{L} \leftarrow \text{compute Eq. 4}$ 

21     Update  $\varepsilon$  while masking a subset of the gradients
22      $\varepsilon \leftarrow \varepsilon + \frac{\partial \mathcal{L}}{\partial \varepsilon} \cdot p$ 

23   end
24   Update counter
25    $i \leftarrow i + 1$ 
26 end

```

low for f_{unbiased} . Formally, the ground truth is computed as

$$\mathbb{E}_{a \sim p(a)} \left[\mathbb{E}_{x, y \sim p(x, y|a)} \left[[\mathbb{1}[y = f(x)|a = a_1] - \mathbb{1}[y = f(x)|a = a_2]] \right] \right], \quad (7)$$

where a represents the attribute, in this case the gender.

J. Out-of-distribution experiment

We test the out-of-distribution (OOD) performance of DiVE with the Symbols dataset [32]. Symbols is an image generator with characters from the Unicode standard and the wide range of artistic fonts provided by the open font community. This provides us to better control on the features present in each set when compared to CelebA. We generate 100K black and white of 32×32 images from 48 characters in the latin alphabet and more than 1K



Figure 8: Sample of the symbols dataset.

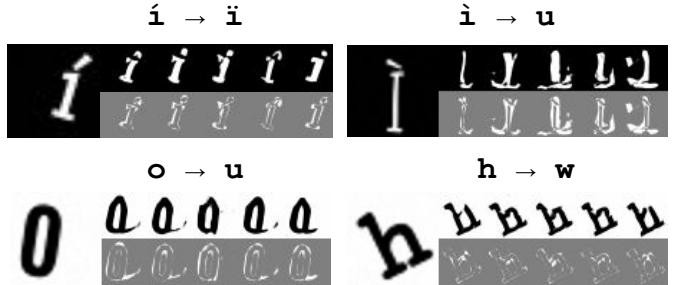


Figure 9: Successful counterfactuals for four different symbols in the OOD regime. Each sample consists of the original image in bigger size, five different counterfactuals generated by DiVE_{FisherSpectral}, and the difference in pixel space with respect to the original image (gray background). The header in each sample indicates the target class, e.g., i, u, w . All the counterfactuals are predicted by the ML model as belonging to the target class and differ from the oracle (*non-trivial*).

fonts (Figure 8). In order to train the VAE on a different disjoint character set, we randomly select the following 32 training characters: $\{a, b, d, e, f, g, i, j, l, m, n, p, q, r, t, y, z, \grave{a}, \acute{a}, \grave{e}, \acute{e}, \grave{e}, \acute{e}, \grave{i}, \acute{i}, \grave{o}, \acute{o}, \grave{u}, \acute{u}, \grave{u}, \acute{u}\}$. Counterfactuals are then generated for the remaining 16 characters: $\{c, h, k, o, s, u, v, w, x, \grave{a}, \acute{i}, \grave{o}, \acute{o}, \grave{u}, \acute{u}\}$.

c	Anonymous Pro, Architects Daughter, BioRhyme Expanded, Bruno Ace, Bruno Ace SC, Cantarell, Eligible, Eligible Sans, Futurespore, Happy Monkey, Lexend Exa, Lexend Mega, Lexend Tera, Lexend Zetta, Michroma, Orbitron, Questrial, Revalia, Stint Ultra Expanded, Syncopate, Topeka, Turret Road
h	Acme, Alatsi, Alfa Slab One, Amaranth, Archivo Black, Atma, Baloo Bhai, Bevan, Black Ops One, Bowlby One SC, Bubblegum Sans, Bungee, Bungee Inline, Calistoga, Candal, CantoraOne, Carter One, Ceviche One, Changa One, Chau Philomene One, Chela One, Chivo, Concert One, Cookbook Title, Days One, Erica One, Francois One, Fredoka One, Fugaz One, Galindo, Gidugu, Gorditas, Gurajada, Halt, Haunting Spirits, Holtwood One SC, Imperial One, Jaldi, Jockey One, Jomhurie, Joti One, Kanit, Kavoon, Keania One, Kenia, Lalezar, Lemon, Lilita One, Londrina Solid, Luckiest Guy, Mitr, Monoton, Palanquin Dark, Passero One, Passion, Passion One, Patua One, Paytone One, Peace Sans, PoetenOne, Power, Pridi, Printed Circuit Board, Rakkas, Rammetto One, Ranchers, Rowdies, Rubik Mono One, Rubik One, Russo One, Saira Stencil One, Secular One, Seymour One, Sigmar One, Skranji, Squada One, Suez One, Teko, USSR STENCIL, Ultra, Unity Titling, Unlock, Vianfot, Wattauchimma, simplicity
k	Abel, Anaheim, Antic, Antic Slab, Armatia, Barriecito, Bhavuka, Buda, Chiflanka, Comfortata, Comic Neue, Cutive, Cutive Mono, Delius, Delius Unicase, Didact Gothic, Duru Sans, Dustismo, Fauna One, Feronia, Flamenco, Gafata, Glass Antiqua, Gothic A1, Gotu, Handlee, IBM Plex Mono Light, IBM Plex Sans Condensed Light, IBM Plex Sans Light, Indra Flower, Josefina Sans Std, Kite One, Manjari, Metrophobic, Miriam Libre, News Cycle, Offside, Overlock, Overlock SC, Panefresco 250wt, Pavaram, Pontana Sans, Post No Bills Colombo Medium, Puritan, Quattrocento Sans, Quicksand, Ruda, Scope One, Snippet, Sulphur Point, Tajawal Light, Text Me One, Thabit, Unkempt, Varta, WebySleek UI, Yaldevi Colombo Light
o	Abhaya Libre, Medium, Abyssinica SIL, Adamina, Alice, Alike, Almendra SC, Amethysta, Andada, Aquifer, Arapay, Asar, Average, Baskerville, Brawler, Cambo, Della Respira, Donegal One, Esteban, Fanwood Text, Fenix, Fjord, GFS Didot, Gabriela, Goudy Bookletter 1911, Habibi, HeadlandOne, IM FELL French Canon, IM FELL French Canon SC, Inika, Jomlhari, Karma, Kreon, Kurale, Lancelet, Ledger, Linden Hill, Linus, Libertine Display, Lustria, Mate, Mate SC, Metamorphous, Milonga, Montaga, New Athena Unicode, OFL Sorts Mill Goudy TT, Ovo, PT Serif Caption, Petrona, Poly, Procione, Quando, Rosario, Sawarabi Mincho, Sedan, SC, Theano Didot, Theano Old Style, Trocchi, Trykker, Uchen
s	Abrial Fattace, Almendra, Arbitus, Asset, Audiowide, BPdotsCondensedDiamond, Bigshot One, Blazium, Bowly One, Brazier Flame, Broken Glass, Bungee Outline, Bungee Shade, Butcherman, CAT Kurier, Cabin Sketch, Catenary Sans, Chango, Charmonnier, Cherry Bomb, Cherry Creek Soda, Chonburi, Codystar, Coiny, Corben, Cousard, Creeperster, CriminalHand, DIN Schablonerschrift Cracked, Devonshire, Diplomat, Diplomat SC, Dr Sugiyama, DraweflicaMin, Eater, Elsie, Emblema One, Faster One, Flavors, Fontliner Swanky, Fredericka the Great, Frijole, Fruktur, Geostar, Geostar Fill, Goblin One, Gravitas One, Hanalei Fill, Irish Grover, Kabinet Fraktur, Katibeh, Knewave, Leckerli One, Lily Script One, Limelight, Linux Biolumin Outline, Linux Biolumin Shadow, Lucien Schoenscrifft CAT, Membra, Metal Mania, Miltonian Tattoo, Modak, Molle, Mortified Drip, Mrs Sheppards, Multivac, New Rocker, Niconne, Nosifer, Notable, Oleo Script, Open Flame, Overhaul, Paper Cuts 2, Peralta, Piedra, Plaster, Poller One, Porter Sans Block, Press Start 2P, Purple Purse, Racing Sans One, Remix, Ringling, Risalut Display, Rye, Sail, Sanidana, Sana, Sarpanch, Schulze Werbekraft, Severely, Shojumaru, Shrikhand, Slackey, Smokum, Sniglet, Sonica One, Sortefax, Spicy Rice, Staline One, SudagnakNo2, Sun Dried, Tangerine, Thickhead, Tippa, Titan One, Tomorrow, Trade Winds, VT23, Vampiro One, Vast Shadow, Video, Vipond Angulus, Wallpoet, Yesteryear, Zilla Slab Highlight
u	Alef, Alegra Sans, Almarai, Archivo, Arimo, Arsenal, Arya, Assistant, Asul, Averia Libre, Averia Sans Libre, Be Vietnam, Belleza, Cambay, Comme, Cousine, DM Sans, Darker Grotesque, DejaVu Sans Mono, Dhanya, Eplexus Sans, Hack, Istok Web, Kofo, Krub, Lato, Liberation Mono, Liberation Sans, Libre Franklin, Luna Sans, Marcellus, Martel Sans, Merriweather Sans, Monda, Mplus 1p, Mukta, Muli, Niramit, Noto Sans, Open Sans, Open Sans Hebrew, PT Sans, PT Sans Caption, Raleway, Rambla, Roboto Mono, Rosario, Rounded Mplus 1c, Sansation, Sarabun, Sarala, Scada, Sintony, Tajawal, Tenor Sans, Voces, Yantramana
v	Alex Toth, Antar, Averia Gruesa Libre, Baloo Bhai 2, Bromite, Butterly Kids, Caesar Dressing, Cagliostro, Capriola, Caveat, Caveat Brush, Chelesa Market, Chewy, Class Coder, Convincing Pirate, Copse, Counterproductive, Courgette, Courier Prime, Covered By Your Grace, Damion, Dear Old Dad, Dekko, Domestic Manners, Dosis, Erica Type, Erika Ormig, Espresso Dolce, Farsan, Finger Paint, Freckle Face, Fuckin Gwenhywfar, Gloria Hallelujah, Gochi Hand, Grand Hotel, Halogen, IM FELL DW Pica, IM FELL DW Pica SC, IM FELL English SC, Itim, Janitor, Junior CAT, Just Another Hand, Just Me Again Down Here, Kalam, Kodchasan, Lacquer, Loren Ipsum, Love Ya Like A Sister, Macondo, Malí, Mansalva, Margarine, Marmelad, Matias, Mogra, Mortified, Nunito, Objective, Oldenburg, Oregon, Pacifico, Pangolin, Patrick Hand, Patrick Hand SC, Peccita, Pianaforma, Pompiere, Rancho, Reenie Beanie, Rock Salt, Rue Boogie, Sacramento, Salsa, Schoolbell, Sedgewick Ave, Sedgewick Ave Display, Shadows Into Light, Short Stack, SirinStencil, Sofadi One, Solway, Special Elite, Sriracha, Stalemate, Sue Ellen Francisco, Sunshiney, Supercomputer, Supermercado, Swanky and Moo Moo, Sweet Spots, Varela Round, Vibur, Walter Turncoat, Warens, Wellfleet, Yellowtail, Zeyada
w	Alata, Alte DIN 1451 Mittelschrift geprägt, Amble, Arvo, Asap, Asap VF Beta, Athiti, Atomic Age, B612, B612 Mono, Barlow, Barlow Semi Condensed, Basic, Blinker, Bree Serif, Cairo, Celio Sans, Chakra Petch, Changa Medium, Cherry Swash, Crete Round, DejaVu Sans, Doppio One, Droid Sans, Elaine Sans, Encode Sans, Encode Sans Condensed, Encode Sans Expanded, Encode Sans Semi Condensed, Encode Sans Wide, Exo 2, Federo, Fira Code, Fira Sans, Fira Sans Condensed, Gontserra, Hammersmith One, Hand Drawn Shapes, Hartmann, Heeo, Hepta Slab, Hind, Hind Kochi, IBM Plex Mono, IBM Plex Mono Medium, IBM Plex Sans, IBM Plex Sans Condensed, Iceland, Josefin Sans, Krona One, Livvic, Madu, Magra, Maven Pro, Maven Pro VF Beta, Mirza, Montserrat, Montserrat Alternates, Myanman Khay, NATS, Nobile, Oxanium, Play, Poppins, Prompt, Prosto One, Proza Libre, Quantico, Red Hat Text, Reem Kufi, Renner*, Righteous, Saira, Saira SemiCondensed, Semi, Share, Signika, Soniano Sans Unicode, Source Code Pro, Source Sans Pro, Spartan, Viga, Yatra One, Zilla Slab
x	Abhaya Libre, Amita, Antic Didone, Aref Ruqaa, Arima Madurai, Bellefair, CAT Childs, CAT Linz, Cardo, Caudex, Cinzel, Cormorant, Cormorant Garamond, Cormorant SC, Cormorant Unicase, Cormorant Upright, Cuprum, Domine, Dustismo Roman, El Messiri, Fahlwang, FogtwoNo5, Forum, Galatia SIL, Gayathri, Gilda Display, GlukMix, Gpuktes, Griffy, Guptier, IBM Plex Serif Light, Inria Serif, Italiana, Judson, Jungo, Libre Caslon Display, Linux Biolumin Capitals, Linux Biolumin Slanted, Linux Libertine Capitals, Lobster Two, Marcellus SC, Martel, Merienda, Modern Antiqua, Montserrat Subrayada, Mountains of Christmas, Mystery Quest, Old Standard TT, Playfair Display, Playfair Display SC, Portmanteau, Prata, Pretzel, Prida36, Quattrocento, Resagni, Risque, Rufina, Spectral SC, Trirong, Viadova Libre, Wes, kawoszeh, okolaks
â	Accuratist, Advent Pro, Archivo Narrow, Aubrey, Cabin Condensed, Convergence, Encode Sans Compressed, Farro, Fira Sans Extra Condensed, Galdeano, Gemunu Libre, Gemunu Libre Light, Geo, Giudea, Homenaje, Iceberg, Liberty Sans, Mohave, Nova Cut, Nova Flat, Nova Oval, Nova Round, Nova Slim, NovaMono, Open Sans Condensed, Open Sans Hebrew Condensed, PT Sans Narrow, Port Ligat Sans, Port Ligat Slab, Pragati Narrow, Rajdhani, Rationale, Roboto Condensed, Ropa Sans, Saira Condensed, Saira ExtraCondensed, Share Tech, Share Tech Mono, Strait, Strong, Tauri, Ubuntu Condensed, Voltaire, Yaldevi Colombo, Yaldevi Colombo Light
i	Aladin, Alegre Sans, Allan, Amarante, Anton, Antonio, Asap Condensed, Astloch, At Sign, Bad Script, Bahiana, Bahianita, Bangers, Barloesius Schrift, Barlow Condensed, Barrio, Bebas Neue, BenchNine, Berlin Email Script, Berlin Email Serif Shadow, Berolina, Butherford Mainzer Fraktur, Biedermeier Text, Big Shoulders Text, Bigelow Rules, Bimbo, Bonbon, Boogaloo, CAT FrankenDeutsch, CAT Liebing Gotisch, Calligraserif, Casa Sans, Chicle, Combe, Contrail One, Crushed, DN Titling, Dagerotyp, Denk One, Digital Numbers, Dorsa, Economica, Eleventh Square, Engagement, Euphoria Script, Ewert, Fette Mikado, Fjalla One, Fluffy, Friedolin, Galada, Germania One, Gianna, Graduate, Hanalei, Jacques Francois Shadow, Jena Gotisch, Jolly Lodge, Julee, Kanzler, Kavivanar, Kazmann Sans, Kelly Slab, Khand, Kotta One, Kranny, Lemonada, Loved by the King, Maiden Orange, Marco, Merienda One, Miltonian, Mouse Memoirs, Nova Script, Odibee Sans, Oswald, Paprika, Pathway Gothic One, Penguin Attack, Pirata One, Pommern Gotisch, Post No Bills Colombo, Princess Sofia, Redressed, Ribeye Marrow, Rum Raisin, Sancreek, Sancretro, Sanitechro, Sevillana, Six Caps, Slim Jim, Smythe, Sofia, Sportrop, Staithches, Stint Ultra Condensed, Tillana, Tulpen One, Underdog, Unica One, UnifrakturMaguntia, Yanone Kaffeesatz
f	Amatic SC, Bellota Text, Bernardo Moda, Blokletters Balpen, Blokletters Potlood, Bubblet Girls, Bungee Hairline, Coming Soon, Crafty Girls, Gemunu Libre, ExtraLight, Give You Glory, Gold Plated, Gruppo, IBM Plex Mono ExtraLight, IBM Plex Mono Thin, IBM Plex Sans Condensed ExtraLight, IBM Plex Sans Condensed Thin, IBM Plex Serif ExtraLight, IBM Plex Serif Thin, Julius Sans One, Jura, Lazebny Computer, Life Savers, Londrina Outline, Londrina Shadow, Mada ExtraLight, Mada Light, Major Mono Display, Megrim, Nixie One, Northampton, Over the Rainbow, Panefresco 1wt, Poiret One, Post No Bills Colombo Light, Raleway Dots, RawengulkSans, Sansation Light, Shadows Into Light Two, Sierra Nevada Road, Slimamif, Snowburst One, Tajawal ExtraLight, Terminal Dosis, Thasadith, The Girl Next Door, Thin Pencil Handwriting, Vibes, Waiting for the Sunrise, Wine One, Yaldevi Colombo ExtraLight
i	Amiri, Buendia, Colaidea, Charis SIL, Crimson Text, DejaVu Serif, Dita Sweet, Droid Serif, EB Garamond, Eagle Lake, Fondamento, Frank Ruhl Libre, Gelasio, Gentium Basic, Gentium Book Basic, Ibarra Real Nova, Junicode, Liberation Serif, Libre Baskerville, Libre Caslon Text, Linux Biolumin, Linux Libertine, Linux Libertine Slanted, Lusitana, Manuale, Merriweather, Noticia Text, PT Serif, Scheherazade, Spectral, Tavaraj, Unna, Vespa Devanagari Libre
ó	ABeeZee, Actor, Aldrich, Alegreya Sans SC, Aleo, Amiko, Andika, Anne Use Your Telescope, Average Sans, Bai Jamjuree, Baumanns, Belgrano, BioRhyme, Biryani, Cabin, Cabin VF Beta, Calling Code, Carme, Carrois Gothic SC, Catamaran, Changa Light, Convincing, Droid Sans Mono, Electroline, Englebert, Fresca, GFS Neohellenic, Imprima, Inconsolata, Inder, Inria Sans, Josefina Slab, K2D, Karla, Kulim Park, Lekton, Lexend Deca, Mako, McLaren, Meera Inital, Molengo, Numans, Orienta, Overpass, Overpass Mono, Oxygen Mono, PT Mono, Panefresco 400wt, Podkova, Podkova VF Beta, Red Hat Display, Rhodium Libre, Ruluko, Sanchez, Santi Trixie, Sawarabi Gothic, Sen, Shanti, Slabo 13px, Slabo 27px, Sometype Mono, Space Mono, Spinnaker, TuffyInfant, TuffyScript, Ubuntu Mono, Varela
ô	Aguafina Script, Akronim, Alex Brush, Arizona, Beth Ellen, Bilbo, Brausepulver, Calligraphitti, Cedarville Cursive, Charm, Clicker Script, Condiment, Cookie, Dancing Script, Dawning of a New Day, Dynalight, Felipa, Great Vibes, Herr Von Muellerhoff, Homemade Apple, Italiano, Jim Nightshade, Kaushan Script, Kristi, La Belle Aurore, League Script, Meddon, Meie Script, Mervale Script, Miama, Miniver, Miss Fajardose, Monsieur La Doulaise, Montez, Mr Bedfort, Mr Dafoe, Mr De Haviland, Mrs Saint Delafied, Norican, Nothing You Could Do, Parisienne, Petit Formal Script, Pinyon Script, Playball, Promocya, Quintessential, Quigley, Rochester, Romanesco, Rouge Script, Ruthie, Satisfy, Seaweed Script, Srisakdi, Vengeance
ü	Aclonica, Alegreya, Alegreya SC, Andada SC, Artifika, Averia Serif Libre, Balthazar, Bentham, Bitter, Cantata One, Crimson Pro, Croissant One, DM Serif Display, Eczar, Emily Candy, Enriqueta, Faustina, Federator, Girassol, Grenze, Halant, Henry Penny, Hermeneus One, IBM Plex Serif, IBM Plex Serif Medium, IM FELL Double Pica, IM FELL Double Pica SC, IM FELL English, IM FELL Great Primer SC, Inknot Antiqua, Jacques Francois, Judges, Kameron, Laila, Lateef, Literata, Lora, Maitree, Markazi Text, Marko One, Monteiro Lobato, Original Surfer, Psicopatologia de la Vida Cotidiana, Radley, Rasa, Ribeye, Robot Slab, Rokkit, Rozha One, Sahitya, Sansita, Simonetta, Source Serif Pro, Spirax, Stardos Stencil, Stoke, Sumana, Uncial Antiqua, Vidaloka, Volkhov, Volkorn, Volkorn SC

Table 7: Font clusters assigned to each character.