arXiv:2012.09301v1 [cs.LG] 16 Dec 2020

Latent-CF: A Simple Baseline for Reverse Counterfactual Explanations

Rachana Balasubramanian, Samuel Sharpe, Brian Barr, Jason Wittenbach, and C. Bayan Bruss

Center for Machine Learning, Capital One

{rachana.balasubramanian,bayan.bruss,brian.barr,samuel.sharpe,jason.wittenbach}@capitalone.com

Abstract

In the environment of fair lending laws and the General Data Protection Regulation (GDPR), the ability to explain a model's prediction is of paramount importance. High quality explanations are the first step in assessing fairness. Counterfactuals are valuable tools for explainability. They provide actionable, comprehensible explanations for the individual who is subject to decisions made from the prediction. It is important to find a baseline for producing them. We propose a simple method for generating counterfactuals by using gradient descent to search in the latent space of an autoencoder and benchmark our method against approaches that search for counterfactuals in feature space. Additionally, we implement metrics to concretely evaluate the quality of the counterfactuals. We show that latent space counterfactual generation strikes a balance between the speed of basic feature gradient descent methods and the sparseness and authenticity of counterfactuals generated by more complex feature space oriented techniques.

1 Introduction

The GDPR provides a "right to be informed" about system functionality in automated decision making processes. Though not explicit in the GDPR, the law in spirit ostensibly encourages those creating algorithmic decision making systems to build trust and increase transparency around these systems [21]. When considering the fair application of artificial intelligence in industrial settings such as finance, it is important to consider the information and agency afforded to those affected by the decisions of the AI system.

In particular, providing the affected party with a means to either change their behavior to ensure a better outcome in the future, or to contest the current outcome. One way to provide this is through counterfactual explanations. Specifically, counterfactual explanations provide a plausible change to a model's input that would result in a change in the model's prediction.

Counterfactual explanations fall within the category of model-agnostic *post-hoc* example based explanations. Counterfactual explanations trace their lineage to philosophy and psychology and are deeply intertwined with analyses of causality. Philosophers proposed the use of counterfactuals as a way to understand causal arguments. If A caused B then if A had not occurred B would not have occurred [11]. Much of the subsequent work in causal inference takes this same approach to counterfactual analysis. Changes to antecedents should result in changes to outcomes otherwise there is no causal relationship [17].

Researchers in psychology have noted the importance of counterfactual thinking in human cognition, with development as early as two years old [4]. Unlike the counterfactuals in philosophy, the functional theory of counterfactual thinking suggests a reverse process. Namely, a person starts with

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

a mismatch between expectations and the present outcome. The counterfactual isn't to understand generic causal relationships, but rather it is goal oriented to ensure a different outcome in the future. In this framing, one starts from the undesirable outcome and works backward until a path opens up to the desirable outcome. This fits more closely with the framework of reverse causal inference [6]. Reverse causal inference starts from a deviation from expectations - typically an anomaly - and performs model checking to understand the causes of the deviation. Because this backward looking process can result in a number of possible antecedent changes, philosophers and psychologists have begun to explore the process by which individuals prune the possible candidate counterfactuals referred to as counterfactual availability [5].

In the domain of model-based counterfactual explorations, the common approach is aligned to the traditional forward looking causal structure. Namely, counterfactual approaches explore alternative inputs that would lead to changes in the model's predictions. The objective of a counterfactual is that given $Y = f(X)$, for a specific x_0 where $f(x_0) = y_1$ generate x_{cf} where $f(x_{cf}) = y_2$. What is clear from this formulation is that there may be many possible paths to generating x_{cf}. As a result, forward looking counterfactuals seek to place constraints on the possible changes both for computational efficiency but also interpretability [21].

In this paper we propose a method for generating counterfactual explanations in-line with the reverse causal inference framework and the functional theory of counterfactual thinking. These frameworks are well-suited to providing those affected by a decision with an explanation, especially when the decision led to an undesired outcome. The explanations produced provide a method of changing behavior to reach a desired goal in such cases.

We refer to our approach as Latent-CF. Given any black-box classifier and a dataset that has been used to train the classifier, Latent-CF can generate counterfactuals. Training a separate autoencoder on the same training data is necessary for our approach. For any data point and desired class confidence of the target counterfactual class, we traverse the latent space of the autoencoder from the original data point's encoded representation until the desired class probability is reached and then use the decoder to generate the corresponding counterfactual. Because the latent space is more compact, starting from the desired outcome and searching for counterfactuals in that space is more computationally efficient than doing so in feature space. In practical terms, the approach is simple but provides a foundation for viewing counterfactual explanations from a more functional perspective compared to existing forward oriented counterfactuals.

In order to evaluate the quality of these counterfactuals we use a framework similar to that of [12] by stating that a counterfactual explanation should be:

1. In distribution
2. Sparse in the number of changes it makes in the features
3. Computationally efficient

These criteria align also to notions of counterfactual availability making them more actionable [5].

We conduct the following experiments to evaluate the quality of the counterfactuals generated by Latent-CF. First, using the criteria above, we compare Latent-CF to approaches that generate counterfactual explanations utilizing gradient descent in feature space varying loss and clipping strategies to encourage the explanations to be in-sample and sparse. Second, we conduct a visual analysis of the interpretability of the counterfactual explanations. Through these experiments, we are able to show that Latent-CF simple method provides a strong baseline to compare current (and future) methods in reverse counterfactual generation.

2 Previous Work

There is a large body of existing work on *post-hoc* explainability, see [14] for an overview. Many methods treat the creation of explanations as a problem of credit assignment - for each input sample, provide the relative importance of the input features to that samples prediction. Early methods had roots in computer vision, where a common approach is to calculate sensitivities and show a heat map depicting which pixels are responsible for an image's classification. These methods, e.g. [19], rely on using a zero information baseline. The need for a baseline can become problematic if applied to tabular or other data where the definition of a baseline may not exist.

The use of local surrogate models, such as LIME [18] and its extensions, can overcome the need for global baselines to be established. However, the perturbation method used does not ensure that the local surrogate is built on in-sample data.

Game theoretic approaches [20], [1], [13] view the credit assignment task as a coalitional game amongst features. These methods baseline a feature's contribution against the average model prediction. Game theoretic approaches are challenged by the exponential number of coalitions that must be evaluated.

Other researchers have developed methods that express the commonality of features the must be present or missing. Dhurandhar et al. [2] try to find perturbations in feature space utilizing an autoencoder reconstruction loss to keep the explanations in sample. Their method seeks to find contrastive explanations, which describe what minimal set of features or characteristics must be missing to explain why it does not belong to an alternate class.

Counterfactual explanations avoid the pitfalls of previous local attribution methods (no need for baselines, no approximation to game theory constructs, no need for universality in features). All they require is a sample in need of explanation and a search method to find the decision boundary. Lash et al. [9] maintain sparsity in their *inverse classification* methods by partitioning the features into mutable and immutable categories, and imposing budgetary constraints on allowable changes to the mutable features. Laugel et al. [10] advocate a sampling approach with their growing spheres method.

Wachter et al. [21] proposed the use of counterfactual explanations to help the individuals impacted by a model based decision why a particular decision was reached, to provide grounds to contest adverse decisions, and to understand what would result in an alternative decision. van Looveren [12] explore loss functions to search for perturbations in feature space and introduce *prototype* loss which encourages the counterfactual to be close to the average representation in latent space of K training samples from the target class. In contrast to our method which searches for a single counterfactual instance, Mothilal et al [15] suggest generating a set of diverse counterfactuals. McGrath et al. [7] used counterfactuals to explain credit application predictions. They introduced weights, based on either global feature importance or nearest neighbors in an attempt to make them more interpretable.

Some recent counterfactual generation techniques search directly in a latent space. Pawelczyk et al. [16] construct a model agnostic technique that sample observations uniformly in l_p spheres around the original point's representation in latent space. Joshi et al. [8] utilize a very similar approach to ours using a VAE or other generative models to learn $p(x)$ and traverse the latent space with gradient descent to find counterfactuals. They include an additional loss term in the feature space to encourage sparse changes. We choose to use the most simple manifestation of latent space generated counterfactuals—a basic autoencoder and loss including only score of a decoded latent representation—to convincingly illustrate the benefits of searching in a smaller representative space.

3 Methods

3.1 Pretrained Models

The classifier is built using Keras in Tensorflow 2. It uses four stacked layers of convolution with a 3x3 kernel size, 2x2 pooling kernels and filter depths = [32, 16, 8, 1] followed by a dense layer with sigmoid activation as the output. It is trained on the standard train/test split of MNIST filtered to only 4's and 9's.

3.2 Latent-CF

In feature space perturbation methods, a sample x_0 is fed through a classifier $f : X \rightarrow [0, 1]$ and iteratively updated until $f(x)$ is close to 0.5 (decision boundary) or some other target probability p where the closeness to the boundary is measured by a loss function $\mathcal{L}$. Approaches in [12] and [2] incorporate an autoencoder in $\mathcal{L}$ to guide the search for counterfactual examples. However, Latent-CF, similar to [8], directly perturbs the latent representation of the encoder, $z = E(x)$, until the probability of the decoded sample, $f(D(z))$, is close to p. Our algorithm is detailed in Algorithm 1 and the architecture of Latent-CF is illustrated in Figure 1.

Algorithm 1: Latent-CF

```
parameter : $p$ probability of target counterfactual class (0.5 for decision boundary), $tol$ tolerance
Input: Instance to explain $x_0$, classifier $f$, encoder $E$, decoder $D$, $\mathcal{L}$ loss function
Output: $x_{cf}$ the counterfactual explanation
Encode sample to latent space $z = E(x_0)$
Calculate $loss = \mathcal{L}(p, f(D(z))$
while $loss > tol$ do
    $z \leftarrow z - \alpha \nabla_z loss$
    Calculate $loss = \mathcal{L}(p, f(D(z))$
$x_{cf} = D(z)$
```

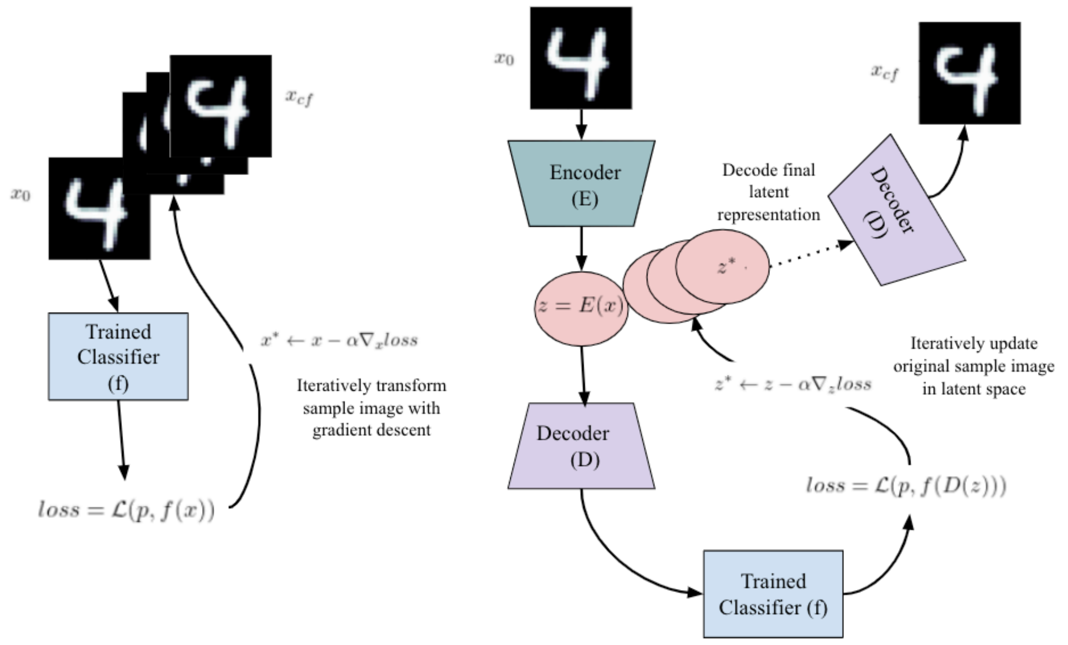

x_0
x_{cf}
Trained Classifier (f)
$x^* \leftarrow x - \alpha\nabla_x loss$
Iteratively transform sample image with gradient descent
$loss = \mathcal{L}(p, f(x))$
x_0
Encoder (E)
$z = E(x)$
Decoder (D)
Decode final latent representation
z^*
Decoder (D)
x_{cf}
$z^* \leftarrow z - \alpha\nabla_z loss$
Iteratively update original sample image in latent space
$loss = \mathcal{L}(p, f(D(z)))$
Trained Classifier (f)

Figure 1: Comparing approaches for counterfactual generation between gradient descent in feature space (left) and Latent-CF gradient descent in the autoencoder latent space (right)

3.3 Baseline Methods

Our first baseline method, *Feature GD*, uses gradient descent to directly perturb the feature space minimizing the ℓ_2 distance to the decision boundary. Two other methods introduce some small changes to make iterative improvements over Feature GD. First, we add feature clipping after every gradient step in *Feature GD + clip* to ensure pixel values stay within the correct domain.

We implement *Feature GD + MAD loss*, which encourages in-sample counterfactuals and sparse changes with Median Absolute Deviation (MAD) scaling loss, as developed by Wachter et al [21]. Instead of the ℓ_2 loss, this method uses the ℓ_1 norm weighted by the inverse median absolute deviation, such that MAD_k of a feature k over the set of points P is as shown in equation 1.

$$MAD_k = median_{j \in P}(|X_{j,k} - median_{l \in P}(X_{l,k})|) \tag{1}$$

This results in a distance metric d for between a synthetic data point x' and original data point x as described by equation 2.

$$d(x, x') = \sum_{k \in F} \frac{|x_k - x'_k|}{MAD_k} \tag{2}$$

This distance metric encourages changing only the pixels that vary in the training set, and punishes changing the border pixels that are rarely nonzero.The architecture for these three methods is illustrated on the left of Figure 1.

We compare a final feature gradient descent method from [12] which we label *Prototype-CF*. The authors include five different loss terms in their objective to achieve desirable properties of their counterfactuals.

$$L = cL_{pred} + \beta L_1 + L_2 + L_{AE} + L_{proto}$$

L_{pred} is designed to encourage counterfactuals of a different class. L_1 and L_2 are combined to form an elastic-net regularizer on the feature perturbations for sparse changes. They include L_{AE} as used by Dhurandhar et al. in [3] to ensure in-sample reconstructions of counterfactuals. Finally, they guide the search for counterfactuals by introducing *prototypes*, which are the average euclidean representation of a class in the latent space defined by the K closest encoded samples to $E(x_0)$. Specifically, for a class i the prototype is defined as

$$proto_i = \frac{1}{K}\sum_{k=1}^{K} E(x_k^i)$$

where $\left\|E(x_k^i) - E(x_0)\right\|_2 \leq \left\|E(x_{k+1}^i) - E(x_0)\right\|_2$. L_{proto}, defined as the distance to the closest prototype, effectively tries to speed up the search for counterfactuals by pushing $x_0 + \delta$ to the closest prototype.

3.4 Evaluation Metrics

In order to evaluate the quality of the counterfactual, we use three evaluation metrics:

1) In-distribution
The counterfactual should not change the image such that proposed features will have a low probability of occurring. We used per-pixel intensity kernel density estimation (KDE) to measure the extent to which our counterfactuals are in-sample. Specifically, we estimate the density over intensity values for each pixel across the target class population. We compute the probability for each pixel intensity given the pixel specific KDE and take the average over all pixels. An example of this can be seen in Figure 2 comparing probabilities of pixels in counterfactuals as well as probabilities of the original pixels.

2) Sparsity
The counterfactual should be sparse in the number of changes it makes in the features. We compute the percent of features that are changed in the generation of the counterfactual.

3) Computational Efficiency
Generating the counterfactual must be computationally efficient. We measure the time it takes to generate a counterfactual.

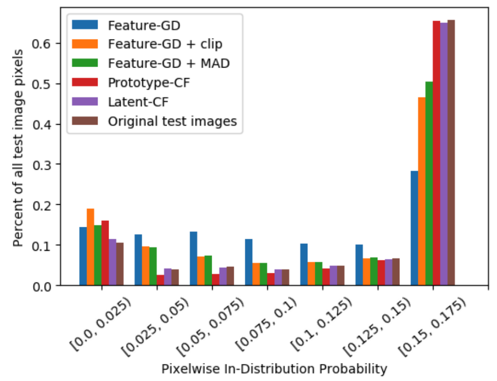


Figure 2: Histogram of Kernel Density Estimate probabilities per pixel computed on original images and counterfactuals. In this case the counterfactual images generated by baseline methods have more low probability pixels than the original images, signifying out of distribution changes.

3.5 Experiments

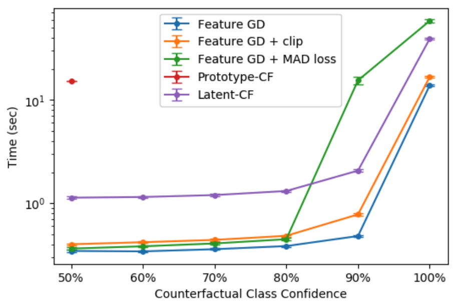
In order to generate a baseline comparison of the counterfactual methods, we analyze the MNIST digit classification dataset. We performed a series of experiments. We first compare each method outlined above as well as the method proposed in [12] on the four criteria proposed. As mentioned in Section 3.1, we use a convolutional neural network for the binary classification task of distinguishing fours and nines. Given this classifier, we generate counterfactuals for fours given nines and vice versa. We also conduct a visual comparison of the counterfactual images generated by each model type, and use these along with our evaluation metrics to draw conclusions about the quality of each method.

Method	**In-Distribution**	**Sparsity (%)**	**Time (s)**
Feature GD	0.098	77.5	**0.346**
Feature GD + clip	0.111	48.7	0.401
Feature GD + MAD loss	0.117	38.4	0.364
Prototype-CF	0.133	**16.8**	15.400
Latent-CF	**0.135**	24.2	1.134

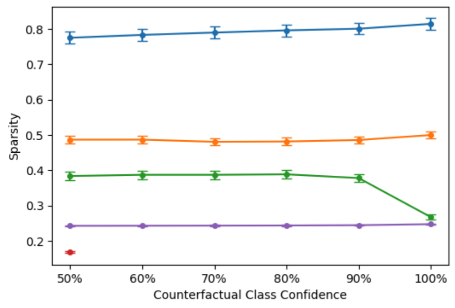
Table 1: Mean and standard deviation for quality metrics given a target counterfactual class probability of 0.5: in-distribution, sparsity, and time for all methods in this study.

4 Results and Discussion

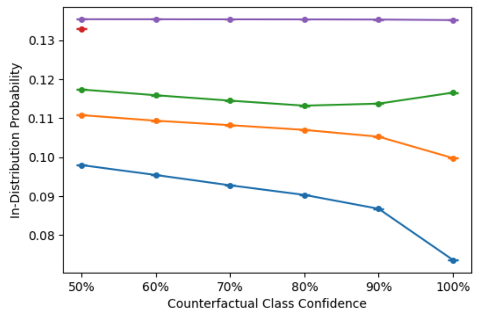
In Table 1, we present a comparison of each method across all of the proposed metrics. Each of these metrics were calculated using a fixed probablity, $p = 0.5$, as the decision boundary. We will investigate each metric in further detail later on, but this serves to point out some immediate trends. First, we can notice that by far Prototype-CF method takes much longer than the Feature GD methods. Secondly, we notice that Latent-CF and the Prototype-CF method change a much smaller percentage of the pixels, making their generated counterfactuals sparser. Finally, Prototype-CF and Latent-CF result in less out-of-distribution changes to the image resulting in in-distribution scores close the original images themselves (0.137).



(a) Counterfactual run time for each method



(b) Fraction of pixels changed by each method

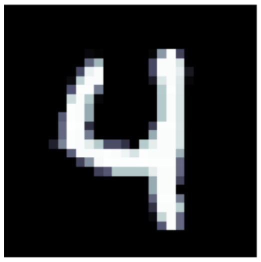


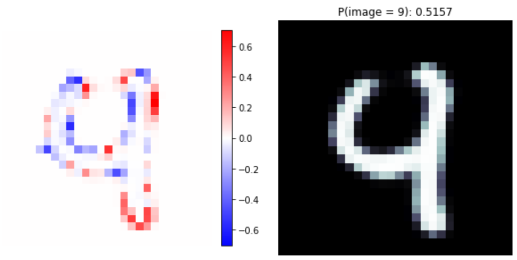
(c) In-distribution probability over all test image pixels

Figure 3: Average and 95% CI of time, sparsity, and in-distribution probability across all test images calculated for various counterfactual confidence levels.

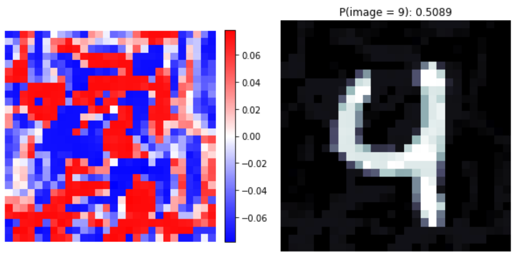
In Figure 3 we show these performance metrics when pushing beyond the decision boundary of 0.5 to higher levels confidence. We only include results for Prototype-CF at 0.5 since their formulation uses a penalty to encourage bounding the confidence from below. Figure 3c shows how the probability of being in sample changes for different confidence values. Models that perform gradient descent in feature space tend to drift more out of sample as the classifier seeks to obtain higher confidence in the counterfactual image, while the Latent-CF method stays relatively stable. We notice that for most Feature GD counterfactuals, the changes are somewhat adversarial in that they make a significant number of perturbations around the border of the image. These changes get larger in magnitude as we target higher confidences, thus decreasing average pixel in-distribution probability.

To get a quantitative evaluation of how each model and method performed in terms of sparsity, we calculated the fraction of the 784 pixels that were changed to get from the original image to the counterfactual image. Figure 3b demonstrates how the proportion of pixels changed varies with increasing confidence.

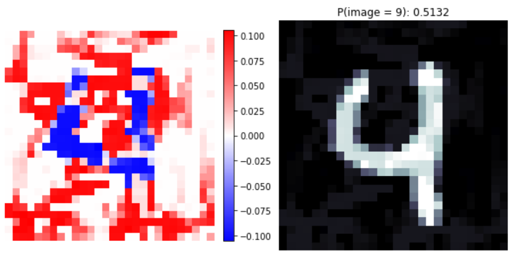

(a) Original Image



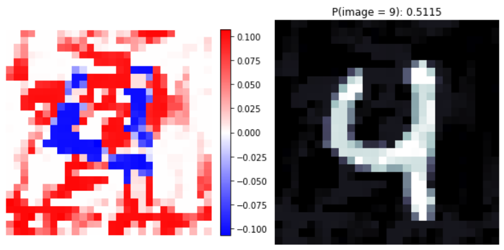
(b) Latent-CF



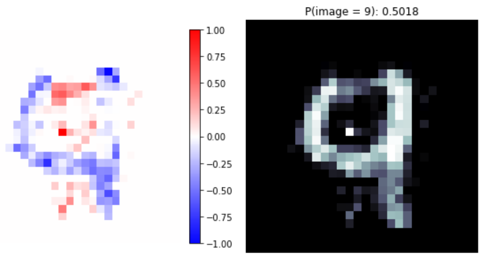
(c) feature gradient descent



(d) feature gradient descent with clipping



(e) MAD loss



(f) Prototype-CF

Figure 4: Visual inspection of counterfactual generation by each of the four methods proposed here. The heat map corresponds to the changes in intensities of each pixel. Note smaller scales for feature gradient descent methods.

Finally, efficiency is important when generating counterfactuals for production systems. Table 1 shows the time in seconds that it took for each model type to generate a counterfactual with a given method. Unsurprisingly, the feature gradient descent methods tend to run the fastest and methods that add additional transformations, architectures, or losses increases the computation necessary and slows down counterfactual generation. We also see these effects across levels of counterfactual confidence in Figure 3a. Latent-CF strikes a balance between all the elements of a quality counterfactual generation method. Latent-CF demonstrates the advantages of Prototype-CF while also resulting in an order of magnitude speed up.

We also provide a visual examination of counterfactuals in Figure 4. We start with an image classified as a 4 (Figure 4a) and generate counterfactual images with each method. We include the counterfactual image and heat map which illustrates the addition or subtraction of luminosity for a pixel in the counterfactual image compared to the original. These are presented for each of the four described methods in Figures 4b to 4f. From the heat maps, we can observe that the initial attempt using feature gradient descent is noisy and does not seem to be confined to any features of the 4. Once we add clipping, we do observe a less noisy heat map, with aspects of the 4 becoming visible, but there are still many noisy changes in surrounding pixels. Using MAD loss results in similar changes as clipping. Finally, we can observe that Latent-CF and Prototype-CF actually begin to modify the shape of the four, adding curvature and closing the top of the of the four as could be expected in a nine. We can see a significant change in the heat map as Latent-CF directly manipulates the shape of the four as changes are concentrated on the border of the four while Prototype-CF makes some extraneous changes inside and outside the main shape.

5 Conclusion

We demonstrate the benefits of performing perturbations in a representative latent space compared to various methods in feature space for counterfactual generation. We show that these methods benefit from sparsity and in-sample perturbations that simple feature based methods lack. Latent based generation also exhibits significant speed up compared to more complex feature based methods designed to remedy simple methods' difficulty generating sparse and in-sample counterfactuals.

Broader Impact

A positive impact of this work is an increased focus in future research on latent perturbation methods to produce higher quality counterfactuals for the benefit of transparency and parties affected by model decisions. Possible negative outcomes of this work is the presence of bias in data and/or classifiers that are not addressed in this approach, and thus, may persist in the counterfactual explanations themselves. The failure of counterfactual systems to provide accurate and fair explanations could lead to counterproductive decisions by affected parties further worsening their standing in the eyes the decision system.

References

[1] A. Datta, S. Sen, and Y. Zick. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 598–617, May 2016.

[2] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 590–601, Red Hook, NY, USA, 2018. Curran Associates Inc.

[3] Amit Dhurandhar, Vijay Iyengar, Ronny Luss, and Karthikeyan Shanmugam. Tip: Typifying the interpretability of procedures, 2018.

[4] Kai Epstude and Neal J Roese. The functional theory of counterfactual thinking. *Personality and social psychology review*, 12(2):168–192, 2008.

[5] Sina Fazelpour. Norms in counterfactual selection. *Philosophy and Phenomenological Research*, 06 2020.

[6] Andrew Gelman and Guido Imbens. Why ask why? forward causal inference and reverse causal questions. Working Paper 19614, National Bureau of Economic Research, November 2013.

[7] Rory Mc Grath, Luca Costabello, Chan Le Van, Paul Sweeney, Farbod Kamiab, Zhao Shen, and Freddy Lécué. Interpretable credit application predictions with counterfactual explanations. *CoRR*, abs/1811.05245, 2018.

[8] Shalmali Joshi, Oluwasanmi Koyejo, Warut Vijitbenjaronk, Been Kim, and Joydeep Ghosh. Towards realistic individual recourse and actionable explanations in black-box decision making systems, 2019.

[9] Michael T. Lash, Qihang Lin, W. Nick Street, and Jennifer G. Robinson. A budget-constrained inverse classification framework for smooth classifiers. In *Proceedings of The IEEE International Conference on Data Mining (ICDM)*, ICDM'17, pages 1184–1193, New Orleans, Louisiana, 2017. IEEE.

[10] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki. Comparison-based inverse classification for interpretability in machine learning. In Jesús Medina, Manuel Ojeda-Aciego, José Luis Verdegay, David A. Pelta, Inma P. Cabrera, Bernadette Bouchon-Meunier, and Ronald R. Yager, editors, *Information Processing and Management of Uncertainty in Knowledge-Based Systems. Theory and Foundations*, pages 100–111, Cham, 2018. Springer International Publishing.

[11] David Lewis. Causation. *The Journal of Philosophy*, 70(17):556–567, 1973.

[12] Arnaud Van Looveren and Janis Klaise. Interpretable counterfactual explanations guided by prototypes, 2019.

[13] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 4765–4774, 57 Morehouse LaneRed HookNYUnited States, 2017. Curran Associates Inc.

[14] Christoph Molnar. *Interpretable Machine Learning*. 2019. `https://christophm.github.io/interpretable-ml-book/`.

[15] Ramaravind Kommiya Mothilal, Amit Sharma, and Chenhao Tan. Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations. *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 607–617, January 2020.

[16] Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. Learning model-agnostic counterfactual explanations for tabular data. *Proceedings of The Web Conference 2020*, Apr 2020.

[17] Judea Pearl. The algorithmization of counterfactuals. *Annals of Mathematics and Artificial Intelligence*, 61(1):29, 2011.

[18] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, New York, NY, USA, 2016. ACM, Association for Computing Machinery.

[19] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3145–3153. JMLR.org, 2017.

[20] Erik Ã… Â trumbelj and Igor Kononenko. An efficient explanation of individual classifications using game theory. *Journal of Machine Learning Research*, 11(1):1–18, 2010.

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