
A APPENDIX

A.1 METRICS FOR COMPARISON

We evaluate the quality of counterfactuals by the metrics taken from literature: 1) time - time required to find a counterfactual for a query sample. Time is one of the most significant strengths of our method. We use the time spent to generate a counterfactual explanation for a query sample on average. Our approach is particularly fast compared to other methods because generating a CE requires a search through a relatively lower dimension and projection through the decoder for no more than a fixed number of interpolated points (search stops when criteria are met). By comparison, iterative search based on gradients or perturbing in the input space can be quite expensive if the search distance is large, the learning rate is low, or the mutable dimensions are high. We do not include the time of training the autoencoder with Gaussian mixture latent space due to the fact that this training time is not trivial. It is a one-time cost and is not related to the scalability of CEs generation for a query sample. Ideally, our method triumphs even more, when the query sample is high dimensional. 2) validity - the percentage of success in generating counterfactuals cross the decision boundary. The goal for generating counterfactuals is that they are counterfactuals, which requires them to be classified as the target class. We evaluate validity by including the pre-trained classifier in the algorithm. Since linear search is performed with the classifier is ended when certain criteria are met for the classifier. We could see, in general, that the performance of this dimension for the three methods is satisfying since all three methods employ the classifier as the end search criteria. 3) proximity - a measure of distance from query sample to counterfactual in latent space. Proximity is used as a constraint to promote the performance of counterfactual generation (?). This metric measures how close a counterfactual is to the query sample it is generated from. Usually, this measure is calculated by the $L2$ norm in the input space. 4) sparsity - a measure of features changed. Sparsity is a common metric in the counterfactual literature (?). Sparsity is a measure of how sparse the change vector is. We calculate the $L1$ norm of the change vector in input space. 5) reconstruction loss - a measure of the CE being close to data manifold (?). We can measure the closeness of a CE to its original dataset by passing a counterfactual through the autoencoder and measuring its reconstruction loss. During the training process of the autoencoder, we try to minimize the reconstruction loss. Intuitively, a sample with a smaller loss should be more in-sample and closer to the original data distribution. An unseen sample should have a larger reconstruction loss concerning the autoencoder.

A.2 ALGORITHMS OF GDL

In GDL, instead of updating parameters of DNN which is commonly applied, we update the input query sample and the latent projection of the query sample correspondingly until it cross the decision boundary and the tolerance is satisfied.

Algorithm 1 Gradient Descent Latent Space

Require: ψ and ϕ the initial parameters of Encoder and Decoder; f classifier; x_b the query sample; tol tolerance; p probability of target counterfactual class (0.5 for decision boundary)

- 1: $z_b \leftarrow \psi(x)$
- 2: $z_t \leftarrow z_b$
- 3: $z_t \xleftarrow{+} \nabla_{z_t} (T - f(\phi(z)))^2$
- 4: $x_t \leftarrow \phi(z_t)$
- 5: **if** $|f(x_t) - T| < tol$ **and** $f(x_t) > p$ **then**
- 6: $x_{cf} = \phi(z_t)$
- 7: **return** x_{cf}
- 8: **end if**

A.3 DETAILS OF AUTOENCODERS AND CLASSIFIER

We tabulate the detailed structures of autoencoders and classifiers during experiment in Section ???. For tabular datasets, to avoid break the differentiation of $(T - f(\phi(z)))^2$, instead of argmax for decoder output, we apply temperature annealing softmax to have a sharper reconstructed sample where we set $t = 0.5$.

Table 1: The network structure of MNIST classifier

Input 28*28
Linear 28*28-128, ReLu
Linear 128-64, ReLu
Linear 64-32, ReLu
Linear 32-1, Sigmoid

Table 2: The network structure of Adult income autoencoder

Encoder	Decoder
Input(1,28,28)	Linear(25,128), Relu
Conv2d(1,8,2), stride 1, ReLu	Linear(128, 29*29*32), unflatten
Conv2d(8,16,2), stride 1, BatchNorm2d 16, ReLu	convtranspose2d(32,16,2),stride 1, batchnorm2d 16, ReLu
Conv2d(16,32,2), stride 1, Leaky ReLu, flatten	convtranspose2d(16,8,2),stride 1, batchnorm2d 8, ReLu
Linear(22912, 128), ReLu,	convtranspose2d(8,1,2), stride 1, sigmoid
Linear(128,15)	output(1,28,28)

The structure of Encoder_u is similar to Encoder but with the last layer of 25 nodes.

Table 3: The network structure of Adult income classifier

Input x (cont_in 6 + cat_in 16/one-hot encoding)
Linear 21-10, ReLu
Linear 10-4, ReLu
Linear 4-2, ReLu
Linear 2-1, Sigmoid

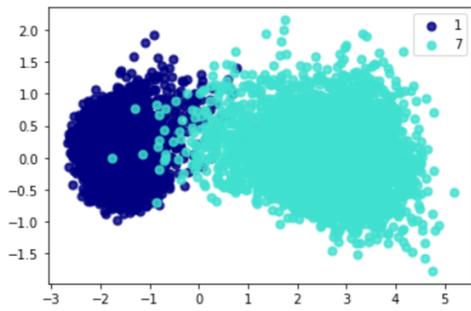
Table 4: The network structure of Adult income autoencoder

Encoder	Decoder	
Input x (cont_in 6/min-max + cat_in 16/one-hot encoding)	Linear 3-6, LeakyReLu	
Linear 21-12, LeakyReLu	Linear 6-12, LeakyReLu	
Linear 12-24, LeakyRelu	Linear 12-24, LeakyReLu	
Linear 24-12, LeakyReLu	Linear 24-5, Tanh	Linear 24-16, softmax
Linear 12-6, LeakyReLu	cont_output 5	cat_output 16
Linear 6-3		

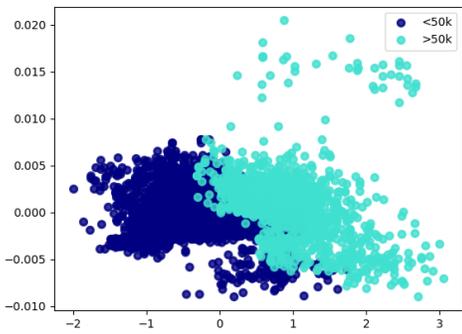
A.4 PCA OF LATENT SPACE

A.5 FURTHER THOUGHTS ABOUT THE COMPARISON RESULTS

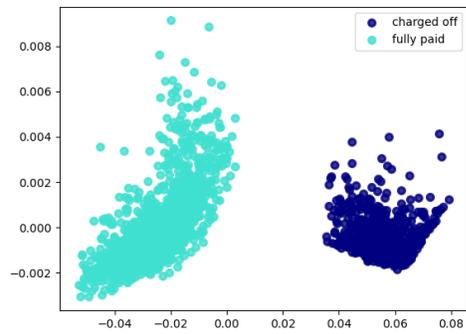
We could see that our method performs better on three dimensions above: counterfactual generation time, validity and reconstruction loss due to its design. We design the architecture based on our three desiderata and in the end, through the experiment, we show that the desiderata are achieved. Our method performs linear interpolation in the latent space instead of optimization in the original space, it has the highest validity, and also because the linear interpolation happens in latent space, we cannot guarantee the sparsity and proximity in the original space, which is stated in Section ??.



(a) MNIST dataset



(b) Adult dataset



(c) Lending Club dataset

Figure 1: PCA of the base and target class in latent space