
A APPENDIX

A.1 LOSS FUNCTION

The goal is to alter the input features of an instance to create a counterfactual that changes the prediction of a machine learning model. The process is guided by balancing prediction change, sparsity of the perturbations, and interpretability. The loss function for generating counterfactual adversarial examples follows an optimization-based approach.

The counterfactual instance, $x_{cf} = x_0 + \delta$, is obtained by applying a perturbation δ to the original instance x_0 . The loss function L to be minimized is constructed as follows:

$$L = L_{\text{pred}} + L_{\text{AE}} + L_{\text{sparse}} + L_{\text{proto}}$$

L_{pred} is designed to ensure the perturbed instance x' is misclassified by the predictive model. It encourages the model to predict a different class for the perturbed instance than for the original. L_{pred} can be defined as:

$$L_{\text{pred}} = \max([f_{\text{pred}}(x_0 + \delta)]_{t_0} - \max_{i \neq t_0} [f_{\text{pred}}(x_0 + \delta)]_i)$$

where $[f_{\text{pred}}(x_0 + \delta)]_i$ is the predicted probability of class i , and t_0 is the original class.

L_{AE} , known as the autoencoder loss, penalizes out-of-distribution instances by measuring the reconstruction error of the perturbed instance using an autoencoder. Moreover, the L_{AE} term aids in ensuring that the generated adversarial example remains realistic by maintaining proximity to the original data manifold. L_{AE} can be defined as:

$$L_{\text{AE}} = \|x_0 + \delta - \text{AE}(x_0 + \delta)\|_2^2$$

This loss helps guide the adversarial instance to stay within the data distribution.

$$L_{\text{sparse}} = L_1 + L_2$$

is an elastic net regularization term that ensures the perturbations are sparse and minimizes the distance between x_0 and x_{cf} . L_1 and L_2 are regularization components aimed at ensuring the sparsity and small magnitude of the perturbation δ . Specifically:

$$L_1 = \|\delta\|_1, \quad L_2 = \|\delta\|_2^2$$

The inclusion of both L_1 and L_2 regularizers, known as the elastic net, balances sparsity with smoothness. The combination of these terms ensures that the perturbation is both small and sparse, preserving the interpretability of the adversarial example while effectively fooling the model.

The prototype loss, L_{proto} , is introduced to speed up convergence and maintain the interpretability of counterfactual instances. It ensures that the perturbation δ aligns with a prototype from the desired class, improving interpretability. Prototypes are derived from an autoencoder representing the mean encoding of the K nearest neighbors in the latent space of the desired class. The prototype term ensures that the generated counterfactual remains within the distribution of the target class.