VERIFIED TRAINING FOR COUNTERFACTUAL EXPLANATION ROBUSTNESS UNDER DATA SHIFT

Anna P. Meyer[*], **Yuhao Zhang**[*], **Aws Albarghouthi & Loris D'Antoni**
Department of Computer Sciences
University of Wisconsin - Madison
Madison, WI, USA
{annameyer, yuhaoz, aws, loris}@cs.wisc.edu

ABSTRACT

Counterfactual explanations (CEs) enhance the interpretability of machine learning models by describing what changes to an input are necessary to change its prediction to a desired class. These explanations are commonly used to guide users' actions, e.g., by describing how a user whose loan application was denied can be approved for a loan in the future. Existing approaches generate CEs by focusing on a single, fixed model, and do not provide any formal guarantees on the CEs' future validity. When models are updated periodically to account for data shift, if the generated CEs are not *robust* to the shifts, users' actions may no longer have the desired impacts on their predictions. This paper introduces VeriTraCER, an approach that *jointly* trains a classifier and an explainer to explicitly consider the robustness of the generated CEs to small model shifts. VeriTraCER optimizes over a carefully designed loss function that ensures the *verifiable* robustness of CEs to local model updates, thus providing deterministic guarantees to CE validity. Our empirical evaluation demonstrates that VeriTraCER generates CEs that (1) are verifiably robust to small model updates and (2) display competitive robustness to state-of-the-art approaches in handling empirical model updates including random initialization, leave-one-out, and distribution shifts.

1 INTRODUCTION

Machine learning models are increasingly used to support decision-making in sectors such as banking, education, social services, and criminal justice. Due to the high stakes of these decision-making settings, and the fact that model internals are often both proprietary and too complex for humans to understand directly, laws such as the GDPR (European Commission, 2018) and ECOA (Consumer Financial Protection Bureau, 2018) require that explanations be offered to users who are subject to these models' predictions. Explanations commonly take the form of counterfactual explanations (CEs), which serve as guidelines for how an input can change in order to receive a different decision in the future. For instance, an individual who is denied a loan may receive a CE that says their application would have been accepted had their salary been $5000 higher. If the applicant wishes to obtain the loan, they can work to increase their salary and then reapply.

However, machine learning models must be periodically updated to account for new data and avoid declining performance due to distribution shift. Even if the CE is valid (i.e., produces the desired prediction) at the time it is generated, there is no guarantee that it will remain valid after one or more routine model updates. So, the individual who successfully raises their salary by the requested $5000 may still be rejected if they reapply for a loan 6 months later, because the model internals shifted.

To preserve the validity of CEs across model shifts, we want to generate CEs that are *robust* to small model changes. Existing work aims to generate robust CEs by increasing the distance from the original input to its CE (Hamman et al., 2023), finding CEs in areas of low Lipschitz constants (Black et al., 2021), or using a minmax objective in the CE generation process (Upadhyay et al., 2021). While all of these methods yield improved robustness over standard training, they fail to provide

[*] Equal contribution

formal guarantees on the CEs' robustness, they require solving *expensive* optimization problems to generate each CE, and they generate CEs with respect to a *fixed* model, even if solving the same optimization problem for a different model – such as one that may be adopted in the future – would yield a better and *still valid* CE.

Our approach, VeriTraCER, fundamentally reframes the problem of generating robust CEs in two ways: first, we consider the *multiplicity set* of similar models, noting that this set likely contains models that may be adopted in the future when accounting for slight data shifts. Second, we adapt existing work, CounterNet, which considers the model training and CE generation processes as a single pipeline (Guo et al., 2023). Specifically, VeriTraCER uses the existing conception of training a model that jointly performs classification and CE generation, but promotes CE robustness by optimizing CE generation over a multiplicity set of classifiers, rather than a single classifier. A key part of how VeriTraCER obtains robustness is to use verified training (Gowal et al., 2018; Zhang et al., 2020) to *deterministically certify* when the predictions and the CEs our model generates are robust to small model changes, as indicated by an l_p-bound on the classifier's parameters, subject to restrictions that hold the original prediction constant. We develop a new variation of verified training, Simul-CROWN, that allows us to obtain tighter bounds than existing approaches. We show that CEs produced by VeriTraCER are not only certifiably robust to small model shifts, but are also robust to other empirical forms of model updates—e.g., training with different random seeds or different data subsets. An added benefit is that CE generation using VeriTraCER is very fast (equivalent to inference speed), with only a modest increase in training time.

In summary, we make the following key contributions: (1) We propose robust CE generation over a multiplicity set of models, (2) we develop a loss function to jointly train an accurate model and a robust and valid CE generator, (3) we design a new verified training algorithm, Simul-CROWN, to soundly overapproximate our loss function during training, and (4) we show that our technique, **Veri**fied **Tra**ining for **CE R**obustness (VeriTraCER), achieves high CE robustness, both to l_p-bounded model shifts and real-world model updates.

2 Related Work

Explainable AI Explanations for model predictions can come from insights about the model (linear regression coefficients, decision tree rules, or feature activations), or more commonly for black-box models, from a post-hoc technique. Post-hoc explanation techniques are typically either feature-based (e.g., Lundberg & Lee (2017); Ribeiro et al. (2016); Simonyan et al. (2013); Smilkov et al. (2017)) and aim to describe which input features are relevant, or counterfactual-based (e.g., Karimi et al. (2020); Looveren & Klaise (2019); Poyiadzi et al. (2020); Ustun et al. (2018); Wachter et al. (2018)) and aim to describe how the original instance needs to change to get a different prediction. In this work, we focus on CEs due to the fact that they provide direct guidance to users. CEs are typically found post-hoc, based on an optimization problem over a fixed model. An exception is the CounterNet training procedure (Guo et al., 2023), which jointly trains a model and CE generator.

Robust explanations "Explanation robustness" refers to multiple phenomena in the literature; the focus can be robustness with respect to the input (i.e., whether perturbing the input yields a similar explanation) or with respect to model changes (i.e., whether changing the model yields a similar prediction and explanation for a fixed input). In this work, we focus on the latter definition.

Various works have explored robust CE generation for specific model classes, such as for tree-based ensembles (Dutta et al., 2022; Forel et al., 2022) or (locally) linear models (Bui et al., 2022; Nguyen et al., 2023). Other work on more complex model classes has shown that it is possible to adversarially change a model to keep identical predictions, but drastically change the associated feature-based explanation (Anders et al., 2020; Heo et al., 2019; Slack et al., 2020). Similarly, a given CE can be invalidated by another equivalently-accurate model in a neural network setting (Hamman et al., 2023). But it is still possible to improve (average) explanation robustness to model shift, such as by increasing the local smoothness of the model (Black et al., 2021; Dombrowski et al., 2022; Meyer et al., 2023; Srinivas et al., 2022). For CEs, increasing the distance to the CE is also commonly thought to increase robustness to model shift (Jiang et al., 2023; Pawelczyk et al., 2020). However, this heuristic does not always work for deep models (Black et al., 2021); instead, the agreement of points in an epsilon-ball around the CE is important (Hamman et al., 2023). However, all of these papers consider CE generation with respect to a fixed model – by contrast, our process jointly trains a model and CE generator in order to obtain robustness. Also, most existing methods only evaluate CE robustness based on empirical model shift. An exception is Jiang et al. (2023), who consider CE robustness to a set of models (via l_p-norm-bounded changes to the parameters of a fixed model). We consider the same l_p-norm bounded setting, but also evaluate our approach on empirical model shifts.

3 Problem Definition

Let f be a neural net classifier that learns its parameters θ_f with a loss function $\mathcal{L}$ and a training set $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$. We will assume that f is a binary classifier, i.e., assigns $\mathbf{x}$ a label $y \in \{0, 1\}$. A distribution shift may occur and yield a new training set $D' = \{\mathbf{x}_i, y_i)\}_{i=1}^k$. We *finetune* f (i.e., train for an additional small number of epochs) or retrain from scratch in the presence of ℓ_2 regularization to yield a shifted model f_{m}. Given f and f_{m} with parameters θ_f and $\theta_{f_{\mathrm{m}}}$, respectively, we measure their distance $\|\theta_f - \theta_{f_{\mathrm{m}}}\|_p$ as the largest ℓ_p parameter distance among all layers (both weight matrices and bias vectors) in f and f_{m}. We make the assumption that the finetuning (or retraining with regularization) process will yield a model f_{m} such that $\|\theta_f - \theta_{f_{\mathrm{m}}}\|_p \leq \delta$ for some small δ. For simplicity, we use a single δ for the l_p bound across all layers. However, our approach easily extends to using different δ values for different layers.

Counterfactual explanations A counterfactual explanation (CE) generator g returns a CE $\mathbf{x}'$ with respect to a function f and an original input $\mathbf{x}$, i.e., $\mathbf{x}' = g(f, \mathbf{x})$. We say that $\mathbf{x}'$ is *valid* if it satisfies $f(\mathbf{x}') \neq f(\mathbf{x})$. The distance between $\mathbf{x}$ and $\mathbf{x}'$ is called *proximity* and should be minimized subject to the validity constraint. Additional constraints (e.g., not changing immutable features like race or gender) can be placed on the CE generation process, e.g., by using a custom distance metric that heavily penalizes changing those features. Our approach is compatible with that type of modification, but we assume an l_1-norm distance metric for simplicity.

Multiplicity set of f Even though f is the result of an optimization process, it likely is not the only model that performs well on the training data due to model multiplicity (D'Amour et al., 2022; Marx et al., 2020). Furthermore, the finetuning or retraining process will yield a distinct model f_{m} that is close to f. We adopt a common assumption from the literature (Jiang et al., 2023; Upadhyay et al., 2021) that the parameter distance between f and f_{m} will be bounded by l_p norms. We define the *multiplicity set* with respect to a model f and an input $\mathbf{x}$ as follows.

Definition 3.1 (Multiplicity set). *Given a model f with parameters θ_f, an input $\mathbf{x}$, an l_p norm, and a bound δ, we define the δ-robust multiplicity set as $\mathcal{M}_{f,\mathbf{x}} = \{f_{\mathrm{m}} \mid f(\mathbf{x}) = f_{\mathrm{m}}(\mathbf{x}) \wedge \|\theta_f - \theta_{f_{\mathrm{m}}}\|_p \leq \delta\}$.*

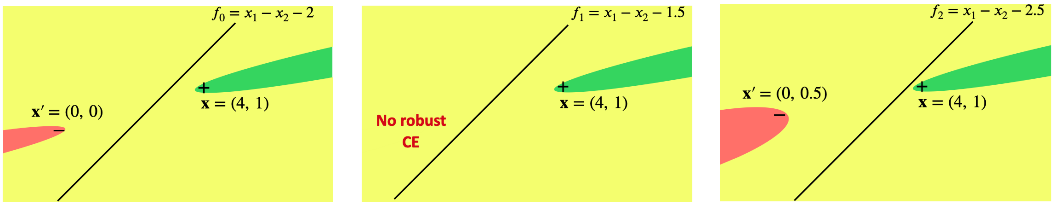
Intuitively, the multiplicity set contains models that have comparable performance to f and have a similar set of model weights. Crucially, it also is intended to contain models f_m that may be the result of model updates due to distribution shift. Note that our definition limits us to models with the same prediction on $\mathbf{x}$. As we are primarily interested in CEs, it no longer makes sense to consider adopting the CE if the prediction changes.

CE robustness *Data shift necessitates that models change over time*, thus creating the risk that CEs generated by a model will be invalidated in the future. We define *CE robustness* as follows.

Definition 3.2 ($\mathcal{M}_{f,\mathbf{x}}$-robustness of a CE). *Given a binary classifier f and an input $\mathbf{x}$, we say that a CE $\mathbf{x}'$ is* robust *if it is a valid CE for all models in the multiplicity set, i.e., if $\forall f_{\mathrm{m}} \in \mathcal{M}_{f,\mathbf{x}}$, $f_{\mathrm{m}}(\mathbf{x}') = 1 - f(\mathbf{x})$.*

Goal Our goal is to devise a training algorithm that yields a model f and a CE generator g such that the CEs generated by g will be $\mathcal{M}_{f,\mathbf{x}}$-robust (i.e., robust to small changes in the model f). In other words, we want to maximize the number of robust CEs that we generate on some dataset (e.g., on the training dataset during training, or on a test or validation dataset post-training). Note that our goal differs from all existing works, which propose algorithms to generates robust CEs according to a fixed model f_{fix}. The following example illustrates how focusing on the model selection process—rather than just on the CE generation technique—can yield better and more robust CEs.

Example 3.1. *Suppose we have an input $\mathbf{x} = (4, 1)$ and the three equally-accurate linear models f_0, f_1, f_2 shown in Figure 1. Considering that these models are equally accurate, we can use CE robustness to small model shifts as a secondary criteria for choosing the best model. To do so, we consider the multiplicity set $\mathcal{M}_{f_i,\mathbf{x}}$ of f_i containing linear models with a bound $\delta = 2$ on the l_∞*



(a) $f_0 = x_1 - x_2 - 2$. The optimal robust CE of $\mathbf{x} = (4, 1)$ is $(0, 0)$.

(b) $f_1 = x_1 - x_2 - 1.5$. There exists no robust CE of $\mathbf{x} = (4, 1)$.

(c) $f_2 = x_1 - x_2 - 2.5$. The opt. robust CE of $\mathbf{x} = (4, 1)$ is $(0, 0.5)$.

Figure 1: Plots of three linear models and their multiplicity sets. The black line shows the original linear model f_i. The green and red regions contain all samples that are robust under the multiplicity set $\mathcal{M}_{f_i,\mathbf{x}}$, receiving predictions $+$ and $-$, respectively. The yellow region corresponds to all samples that are not robust under the multiplicity set $\mathcal{M}_{f_i,\mathbf{x}}$.

norm. In Figure 1, the yellow regions represent all samples that are not robust to $\mathcal{M}_{f_i,\mathbf{x}}$. The optimal robust CE of $\mathbf{x}$ on f_0 is $\mathbf{x}'_0 = (0, 0)$. However, $\mathbf{x}$ has no robust CE on f_1, and its optimal robust CE on f_2 is $\mathbf{x}'_2 = (0, 0.5)$. In other words, if $f_{\text{fix}} = f_1$, then no CE generation algorithm g will be able to find a robust counterfactual. But if $f_{\text{fix}} = f_2$, rather than f_0, we can find a better robust CE (since $\mathbf{x}'_2$ is closer to $\mathbf{x}$ than $\mathbf{x}'_0$ is). So, if we train f and g jointly, we can try to end up with a model closer to f_2: that is, a model that performs well and also yields robust CEs that are of high-quality (i.e., close to the original sample).

4 Approach

Our approach, VeriTraCER, is a training algorithm that takes a training set D as input and outputs a model f and a CE generator g such that the CEs generated by g have a high rate of $\mathcal{M}_{f,\mathbf{x}}$-robustness. Devising such a training algorithm requires us to solve the following two challenges. First, training a model to produce robust explanations requires reasoning in tandem about the accuracy of the model f and the validity, quality, and robustness of the CEs produced by the CE generator g (Section 4.1). Second, we need to be able to *soundly overapproximate* our robust loss function, as using gradient descent to approximate the loss will be both inefficient and may overlook some $f_{\text{m}} \in \mathcal{M}_{f,\mathbf{x}}$ (Section 4.2).

4.1 Training the model f and the CE generator g in tandem

To intertwine model training and robust CE generation, we build off of CounterNet (Guo et al., 2023), which jointly learns a model f and a (non-robust) CE generator g. Intuitively, we have two loss functions, $\mathcal{L}_f$ and $\mathcal{L}_g$, that are used to optimize f and g, respectively. In each training epoch, there are three steps. First, we generate CEs $\mathbf{x}'_i = g(f, \mathbf{x}_i)$ according to the current weights of f and g. Next, we optimize the parameters of f using gradient descent (WRT f) on $\mathcal{L}_f$, and finally, we optimize the parameters of g using gradient descent (WRT g) on $\mathcal{L}_g$.

We denote the output of model f after the final sigmoid layer but before discretization as $\bar{f}$. The function $\bar{f}(\mathbf{x}) \in [0, 1]$ is frequently employed in loss functions for improved optimization compared to the hard label $f(\mathbf{x}) \in \{0, 1\}$. We define helper loss functions $\mathcal{L}_{\text{A}}$, $\mathcal{L}_{\text{V}}$, and $\mathcal{L}_{\text{Q}}$ to promote accuracy, CE validity, and CE quality (i.e., proximity to the original sample $\mathbf{x}_i$), respectively. All three losses were used in the original CounterNet loss, and can be any binary loss function, e.g., MSE. $\mathcal{L}_{\text{R}}$ is our proposed RobustCE loss, and aims to measure the robustness of a CE $\mathbf{x}'$ with respect to $\mathcal{M}_{f,\mathbf{x}}$ (as defined by hyperparameters δ and p, which are omitted in the equations below for brevity). This loss function should capture the worst classifier $f_{\text{m}} \in \mathcal{M}_{f,\mathbf{x}}$, i.e., the model that does most poorly on the CE $\mathbf{x}'$. We define $\mathcal{L}_{\text{R}}(\mathbf{x}, \mathbf{x}', \theta_f) = \max_{f_{\text{m}} \in \mathcal{M}_{f,\mathbf{x}}} \mathcal{L}_{\text{MSE}}(\bar{f}_{\text{m}}(\mathbf{x}'), 1 - y)$. With those definitions in mind, we define $\mathcal{L}_f$ and $\mathcal{L}_g$ as follows:

$$\mathcal{L}_f(\mathbf{x}_i, y_i, \mathbf{x}'_i, \theta_f) = \lambda_1 \mathcal{L}_{\text{A}}(\bar{f}(\mathbf{x}_i), y_i) + \lambda_2 \mathcal{L}_{\text{R}}(\mathbf{x}_i, \mathbf{x}'_i, \theta_f) \tag{1}$$

$$\mathcal{L}_g(\mathbf{x}_i, y_i, g(f, \mathbf{x}_i), \theta_f) = \lambda_3 \mathcal{L}_{\text{Q}}(\mathbf{x}_i, g(f, \mathbf{x}_i)) + \lambda_4 \mathcal{L}_{\text{V}}(\bar{f}(g(f, \mathbf{x}_i)), 1 - y_i) + \lambda_2 \mathcal{L}_{\text{R}}(\mathbf{x}_i, g(f, \mathbf{x}_i), \theta_f) \tag{2}$$

4.2 Computing the RobustCE loss

Exactly minimizing $\mathcal{L}_{\mathrm{R}}$ is expensive: we either need to consider infinitely many reasonable models in $\mathcal{M}_{f,\mathbf{x}}$ or we need to rely on expensive gradient descent to approximate the model f_{m} with no guarantee that it is the worst-case model. To address this, we use abstract interpretation (Cousot & Cousot, 1977) to efficiently compute an upper bound $\mathcal{L}^{\sharp}_{\mathrm{R}}(\mathbf{x}, \mathbf{x}', \theta_f)$ on the RobustCE loss. This upper bound, when minimized and sufficiently tight, ensures a reduction in the RobustCE loss simultaneously. The challenge of overapproximation is how to compute the upper bound $\mathcal{L}^{\sharp}_{\mathrm{R}}(\mathbf{x}, \mathbf{x}', \theta_f)$ as tightly as possible: overly loose bounds may hamper prediction accuracy, CE validity, and CE quality, without meaningfully enhancing CE robustness.

In Section 4.2.1, we show how to compute $\mathcal{L}^{\sharp}_{\mathrm{R}}$ using two existing abstract interpretation techniques, IBP (Gowal et al., 2018) and CROWN-IBP (Zhang et al., 2020). Then, in Section 4.2.2 we introduce a new technique that yields tighter upper bounds on the robust loss while maintaining efficiency similar to CROWN-IBP.

4.2.1 Overapproximating the RobustCE loss using existing techniques

Interval bound propagation (IBP) allows us to evaluate a function on an infinite set of inputs represented as a hyperrectangle in $\mathbb{R}^n$. We will use an interval $\mathbf{z}^{\sharp} = [\mathbf{z}^L, \mathbf{z}^U]$, where $\mathbf{z}^L, \mathbf{z}^U \in \mathbb{R}^n$ and $\forall 1 \leq i \leq n$, $\mathbf{z}^L_i \leq \mathbf{z}^U_i$, to denote the set of all n-dimensional vectors whose i-th element is between $\mathbf{z}^L_i$ and $\mathbf{z}^U_i$, inclusive. Previous work (Gowal et al., 2018; Zhang et al., 2021) has used IBP to overapproximate the worst-case loss of test-time adversarial attacks. These works look at perturbations of an input $\mathbf{x}$, i.e., an infinite set of possible $\mathbf{x}$, as evaluated on a fixed model. By contrast, we want to use IBP to overapproximate the infinitely many models in $\mathcal{M}_{f,\mathbf{x}}$.

To use IBP, we will relax the definition of $\mathcal{M}_{f,\mathbf{x}}$ to $\mathcal{M}_f$ by removing the requirement that the classification of $\mathbf{x}$ remains constant, i.e., $\mathcal{M}_f = \{f_{\mathrm{m}} \mid \|\theta_f - \theta_{f_{\mathrm{m}}}\|_p \leq \delta\}$. After relaxation, the components of θ_f (the weights and biases in linear layers) can be overapproximated by intervals. Therefore, we apply IBP to overapproximate the RobustCE loss under the multiplicity $\mathcal{M}_f$ using interval arithmetic.

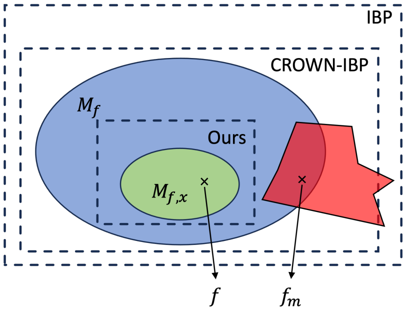


Figure 2: Our approach Simul-CROWN achieves a tighter overapproximation than IBP and CROWN-IBP because the latter techniques include portions of the red region where the CE is not robust.

CROWN-IBP (Zhang et al., 2020) achieves tighter bounds than IBP by incorporating a backward tightening technique employed in α-CROWN (Zhang et al., 2018; Singh et al., 2019). In the evaluation of a function $f_{\theta^{\sharp}}$ whose parameters $\theta^{\sharp}$ are intervals, CROWN-IBP finds $\boldsymbol{\alpha}^l, \boldsymbol{\alpha}^u, \beta^l, \beta^u$ such that $\sigma(\boldsymbol{\alpha}^l\theta + \beta^l) \leq \bar{f}_{\theta}(\mathbf{x}) \leq \sigma(\boldsymbol{\alpha}^u\theta + \beta^u)$ for all $\theta \in \theta^{\sharp}$. Then, these equations ($\boldsymbol{\alpha}^l\theta + \beta^l$ and $\boldsymbol{\alpha}^u\theta + \beta^u$) serve to bound $\bar{f}$, allowing us to do IBP with tighter – yet still sound – bounds. Similar to the adaption of IBP, we need to relax the definition of $\mathcal{M}_{f,\mathbf{x}}$ to $\mathcal{M}_f$ for CROWN-IBP because it cannot deal with the additional requirement of prediction robustness that $\mathcal{M}_{f,\mathbf{x}}$ requires. After this relaxation, the parameters in all layers (i.e., the weight matrices and bias vectors), can be overapproximated into intervals. When applying CROWN-IBP to overapproximate the RobustCE loss, the interval $\theta^{\sharp}$ becomes the intervals of parameters in all layers, and the $\boldsymbol{\alpha}^l, \boldsymbol{\alpha}^u, \beta^l, \beta^u$ are computed based on a specific input $\mathbf{x}$ and the lower and upper bounds of parameters, i.e., θ^L and θ^U.

Theorem 4.1 (Soundness and Tightness). *For any $\mathbf{x}$, $\mathbf{x}'$, and f, the CROWN-IBP-overapproximated loss $\mathcal{L}^{\sharp\mathrm{CIBP}}_{\mathrm{R}}(\mathbf{x}, \mathbf{x}', \theta_f)$ is an upper bound of the RobustCE loss $\mathcal{L}_{\mathrm{R}}(\mathbf{x}, \mathbf{x}', \theta_f)$ and a lower bound of the IBP-overapproximated loss $\mathcal{L}^{\sharp\mathrm{IBP}}_{\mathrm{R}}(\mathbf{x}, \mathbf{x}', \theta_f)$. Formally,*

$$\mathcal{L}^{\sharp\mathrm{IBP}}_{\mathrm{R}}(\mathbf{x}, \mathbf{x}', \theta_f) \geq \mathcal{L}^{\sharp\mathrm{CIBP}}_{\mathrm{R}}(\mathbf{x}, \mathbf{x}', \theta_f) \geq \mathcal{L}_{\mathrm{R}}(\mathbf{x}, \mathbf{x}', \theta_f)$$

Example 4.1. *Consider an interval linear layer with weight matrix $W^{\sharp} = ([-1, 3], [-3, 1])$ and bias scalar $b^{\sharp} = [-4, 0]$. For an input $\mathbf{x} = (-4, -1)$, CROWN-IBP concatenates $W^{\sharp}$ and $b^{\sharp}$ to form $\theta^{\sharp} = ([-1, 3], [-3, 1], [-4, 0])$ and computes $\boldsymbol{\alpha}^l = \boldsymbol{\alpha}^u = (\mathbf{x}; 1) = (-4, 1, 1)$ (note that we append 1 to $\mathbf{x}$ as the multiplicative factor for $b^{\sharp}$) and $\beta^l = \beta^u = 0$. Then we can symbolically* *compute the output based on $\boldsymbol{\alpha}^{l^T}\theta + \beta^l$ (or $\boldsymbol{\alpha}^{u^T}\theta + \beta^u$ because the two are the same), i.e., $(-4, -1, 1)^\top([-1, 3], [-3, 1], [-4, 0])$. To compute the concrete upper bounds, we apply interval arithmetic to the vector product and get $[-17, 7]$. Note that IBP and CROWN-IBP yield identical lower and upper bounds for this linear model because it has only one layer.*

However, the bounds computed by CROWN-IBP are still loose due to the relaxation from $\mathcal{M}_{f,\mathbf{x}}$ to $\mathcal{M}_f$ as illustrated in the following example.

Example 4.2. *Consider the same interval linear model as in Example 4.1. Note that the midpoint of this interval network corresponds to our "true" f, i.e., $W = (1, -1)$ and $b = -2$. Let $\bar{f}(\mathbf{x}) = \sigma(W\mathbf{x} + b)$. For the input $\mathbf{x} = (4, 1)$ and its CE $\mathbf{x}' = (-4, -1)$, we have $\bar{f}(\mathbf{x}) > 0.5$ and $\bar{f}(\mathbf{x}') < 0.5$. IBP and CROWN-IBP are unable to prove that the CE $\mathbf{x}'$ is robust even though $\mathcal{M}_{f,\mathbf{x}}$ does not intersect with the "unsafe" red region shown in Figure 2. To see this limitation, let us consider a model f_{m} with $W' = (-1, -3) \in W^\sharp$ and $b' = 0 \in b^\sharp$ that is in $\mathcal{M}_f$ but outside $\mathcal{M}_{f,\mathbf{x}}$ (the output of $\bar{f}_{\mathrm{m}}(\mathbf{x})$ is $\sigma(-7) < 0.5$). f_{m} has an output of $\sigma(W'\mathbf{x}' + b') = \sigma(7) > 0.5$ on the CE $\mathbf{x}'$ meaning that the CE is invalid on f_{m}. This limitation arises from the relaxation from $\mathcal{M}_{f,\mathbf{x}}$ to $\mathcal{M}_f$. Our approach (Section 4.2.2) overcomes this limitation by overapproximating $\mathcal{M}_{f,\mathbf{x}}$ instead of its relaxation $\mathcal{M}_f$ and is able to prove that $\mathbf{x}'$ is $\mathcal{M}_{f,\mathbf{x}}$-robust.*

4.2.2 OVERAPPROXIMATION THE ROBUSTCE LOSS USING SIMUL-CROWN

The primary challenge faced by IBP and CROWN-IBP in handling $\mathcal{M}_{f,\mathbf{x}}$ is their inability to *simultaneously* reason about the overapproximation of $f(\mathbf{x})$ and $f(\mathbf{x}')$. In this section, we present our approach, Simul-CROWN, which addresses this challenge to achieve a more precise overapproximation of the RobustCE loss.

To simultaneously overapproximate $f(\mathbf{x})$ and $f(\mathbf{x}')$, we first reframe the definition of the RobustCE loss, $\mathcal{L}_{\mathrm{R}} = \max_{f_{\mathrm{m}} \in \mathcal{M}_{f,\mathbf{x}}} \mathcal{L}_{\mathrm{MSE}}(\bar{f}_{\mathrm{m}}(\mathbf{x}'), 1 - \hat{y})$. Namely, if $\hat{y} = 1$, $\mathcal{L}_{\mathrm{R}}$ is maximized when the worst-case classifier is $[\arg\max_{f_{\mathrm{m}} \in \mathcal{M}_f} \bar{f}_{\mathrm{m}}(\mathbf{x}') \quad \text{s.t. } \bar{f}_{\mathrm{m}}(\mathbf{x}) \geq 0]$. However, if $\hat{y} = 0$, $\mathcal{L}_{\mathrm{R}}$ is maximized when the worst-case classifier is $[\arg\max_{f_{\mathrm{m}} \in \mathcal{M}_f} \bar{f}_{\mathrm{m}}(\mathbf{x}') \quad \text{s.t. } \bar{f}_{\mathrm{m}}(\mathbf{x}) \geq 0]$. Using this insight, we apply CROWN-IBP to obtain lower and upper bounds in $\mathcal{M}_f$ for each training instance $\mathbf{x}_i$. For all parameters $\theta \in \theta^\sharp$, we have $\sigma(\boldsymbol{\alpha}_i^l\theta + \beta_i^l) \leq \bar{f}_\theta(\mathbf{x}) \leq \sigma(\boldsymbol{\alpha}_i^u\theta + \beta_i^u)$ and $\sigma(\boldsymbol{\mu}_i^l\theta + \nu_i^l) \leq \bar{f}_\theta(\mathbf{x}) \leq \sigma(\boldsymbol{\mu}_i^u\theta + \nu_i^u)$. We compute $\boldsymbol{\alpha}_i$ and β_i based on $\mathbf{x}_i$ and $\theta^\sharp$, while $\boldsymbol{\mu}_i$ and ν_i are computed based on $\mathbf{x}_i'$ and $\theta^\sharp$.

Theorem 4.2 (Overapproximation by Simul-CROWN)**.** *Conditional on the value of $\hat{y}$, the optimal value of $\mathcal{L}_{\mathrm{R}}$ can be upper bounded by the solution to one of the following cases,*

$$\begin{cases} t_{\hat{y}=1} = \max\limits_{\theta \in \theta^\sharp} \ \boldsymbol{\mu}_i^u\theta + \nu_i^u & \textit{s.t. } \boldsymbol{\alpha}_i^u\theta + \beta_i^u \geq 0, \\ t_{\hat{y}=0} = \min\limits_{\theta \in \theta^\sharp} \ \boldsymbol{\mu}_i^l\theta + \nu_i^l & \textit{s.t. } \boldsymbol{\alpha}_i^l\theta + \beta_i^l \leq 0, \end{cases} \tag{3}$$

That is, if $\hat{y} = 1$, we have $\mathcal{L}_{\mathrm{R}} \leq \mathcal{L}_{\mathrm{MSE}}(\sigma(t_{\hat{y}=1}), 0)$, and if $\hat{y} = 0$, $\mathcal{L}_{\mathrm{R}} \leq \mathcal{L}_{\mathrm{MSE}}(\sigma(t_{\hat{y}=0}), 1)$.

We can use a solver to address each case in Equation (3) by encoding it into a linear programming (LP) problem. However, solving an LP for each training batch is time-consuming and breaks the gradient information required to optimize the RobustCE loss. Appendix A introduces a greedy algorithm (Algorithm 2) that solves Equation (3) in $O(n \log n)$ time, where n is the size of parameters, and can be implemented in PyTorch preserving the gradient information.

Theorem 4.3 (Soundness and Tightness)**.** *For any $\mathbf{x}$, $\mathbf{x}'$, and f, the overapproximated loss $\mathcal{L}_{\mathrm{R}}^{\sharp\mathrm{Ours}}(\mathbf{x}, \mathbf{x}', \theta_f)$ is an upper bound of the RobustCE loss $\mathcal{L}_{\mathrm{R}}(\mathbf{x}, \mathbf{x}', \theta_f)$ and a lower bound of the CROWN-IBP-overapproximated loss $\mathcal{L}_{\mathrm{R}}^{\sharp\mathrm{CIBP}}(\mathbf{x}, \mathbf{x}', \theta_f)$. Formally,*

$$\mathcal{L}_{\mathrm{R}}^{\sharp\mathrm{CIBP}}(\mathbf{x}, \mathbf{x}', \theta_f) \geq \mathcal{L}_{\mathrm{R}}^{\sharp\mathrm{Ours}}(\mathbf{x}, \mathbf{x}', \theta_f) \geq \mathcal{L}_{\mathrm{R}}(\mathbf{x}, \mathbf{x}', \theta_f)$$

Note that $\mathcal{L}_{\mathrm{R}}^{\sharp\mathrm{Ours}}(\mathbf{x}, \mathbf{x}', \theta_f) = \mathcal{L}_{\mathrm{R}}^{\sharp\mathrm{CIBP}}(\mathbf{x}, \mathbf{x}', \theta_f)$ when for all $1 \leq i \leq n$, $\boldsymbol{\mu}_i^u\boldsymbol{\alpha}_i^u > 0$ if $\hat{y} = 1$ (or $\boldsymbol{\mu}_i^l\boldsymbol{\alpha}_i^l > 0$ if $\hat{y} = 0$).

The following example shows that Simul-CROWN is tighter than the CROWN-IBP-overapproximated loss.

Example 4.3. *Consider the same linear model given in Example 4.2, for the original input* $\mathbf{x} = (4, 1)$ *and the corresponding CE* $\mathbf{x}' = (-4, -1)$*, we have the following optimization problem according to Equation* (3)*.*

$$\max_{\theta \in \theta^\sharp} -4z_1 - z_2 + z_3 \quad \textit{s.t. } 4z_1 + z_2 + z_3 \geq 0, \textit{ where } \theta^L = [-1, -3, -4], \theta^U = [3, 1, 0]$$

According to Algorithm 2, the optimal value 0 *is obtained when* $\theta = [-\frac{1}{4}, 1, 0]$*. In other words, the output of the worst-case classifier in* $\mathcal{M}_{f,\mathbf{x}}$ *is* $\sigma(0)$ *computed by Simul-CROWN. This upper bound is tighter than the ones computed by IBP and CROWN-IBP (*$\sigma(7)$ *in Example 4.2).*

5 Experimental Evaluation

In this section, we present our evaluation of VeriTraCER. First, we describe the experiment setup (Section 5.1), and next we show results (Section 5.2) on δ-robustness, cross-model validity, and CE quality.

5.1 Experimental setup

Datasets We perform our evaluation on two *real-world distribution shift* datasets, Cardiotocography (CTG) (Campos & Bernardes, 2010) and WHO (Rajarshi, 2017), as well as three other datasets commonly used in the robust counterfactual literature, Home Equity Line of Credit (HELOC) (FICO, 2018), Taiwanese Credit (TC) (Yeh & Lien, 2009), and Open University Learning Analytics (OULA) (Kuzilek et al., 2017). CTG aims to predict whether a fetal cardiotocogram is healthy, WHO aims to predict whether a country's life expectancy is above the median, HELOC aims to predict the likelihood that home buyers will repay a loan within 2 years, TC aims to predict default on their credit card payments, and OULA aims to predict whether a student will pass an online class. Additional details including dataset composition and preprocessing steps such as binarization and distribution shift construction are available in the appendix.

Metrics Given an $\mathbf{x}$ and a corresponding CE $\mathbf{x}'$, δ-robustness is satisfied when we can certify that $\mathbf{x}'$ obeys $f_{\mathrm{m}}(\mathbf{x}') = f(\mathbf{x}')$ for all $f_{\mathrm{m}} \in \mathcal{M}_{f,\mathbf{x}} = \{f_{\mathrm{m}} \mid f(\mathbf{x}) = f_{\mathrm{m}}(\mathbf{x}) \wedge \|\theta_{f_{\mathrm{m}}} - \theta_f\|_p \leq \delta\}$ using Simul-CROWN. We dynamically determine distinct δ_i values for each layer (see the appendix for details); for simplicity, we will continue to write δ-robustness as if δ is a constant. By contrast, $\mathbf{x}'$ exhibits cross-model validity with respect to two models f and f_{m} when $f(\mathbf{x}') = f_{\mathrm{m}}(\mathbf{x}')$. In practice, we compute empirical cross-model validity across sets of models trained with one of three variations: different random initialization (RI) for training f and f_{m}, different training datasets constructed by randomly removing 1% of the data (LOO) for f and f_{m}, or different training datasets under data shift (DS) for f and f_{m}. We refer to the *δ-robustness rate* as the fraction of data points in a test set that exhibit δ-robustness, and similarly for *cross-model validity rate*. As a secondary metric, we consider how feasible the counterfactual is to implement through its proximity, i.e., the l_1 distance between $\mathbf{x}$ and $\mathbf{x}'$. We consider two other distance metrics in the appendix.

Baseline comparisons We compare our approach with the following existing robust counterfactual generation methods: Counternet (CN) (Guo et al., 2023) without our robustifying modifications; ROAR (Upadhyay et al., 2021), which finds robust counterfactuals by using adversarial training in the counterfactual generation process; and SNS (Black et al., 2021), which finds CEs in regions with a low Lipschitz constant. For ROAR and SNS, we train NNs with the same architecture as the predictor part of CounterNet; we optimize this model using a loss function that solely prioritizes accuracy. We implement all three techniques (IBP, CROWN-IBP, Simul-CROWN) from Section 4.2. We include results for Simul-CROWN here and put results for the others in the ablation study in the appendix.

Experimental procedure To evaluate δ-robustness, we select layer-specific δ_i (see appendix for details) and perform training and all evaluation for that setting. To evaluate cross-model validity for RI and LOO, we train 10 models, generate CEs for each, and report the average fraction of these CEs that remain valid across the other 9 models. To evaluate cross-model validity for DS, we train a model on the original data, then finetune for a small number (typically 20) additional epochs on the new data to obtain the shifted model.

5.2 Results

Table 1: Fraction of samples that are pair-wise cross-model validity across 10 model trained with different random initializations (RI) or different segments of data randomly removed (leave-one-out, or LOO). Standard deviations are in parenthesis. Best result is in **bold** and second-best result is underlined.

	Random Initialization			Leave-One-Out		
	HELOC	TC	OULA	HELOC	TC	OULA
VeriTraCER	**0.93** (0.03)	0.88 (0.14)	**0.94** (0.05)	0.86 (0.14)	0.97 (0.01)	0.96 (0.05)
CN	0.92 (0.07)	0.82 (0.26)	0.92 (0.07)	0.91 (0.07)	0.96 (0.02)	**0.97** (0.01)
ROAR	0.88 (0.02)	0.44 (0.25)	0.93 (0.05)	0.88 (0.02)	0.88 (0.01)	0.95 (0.04)
SNS	0.90 (0.04)	**0.98** (0.02)	0.80 (0.11)	**0.94** (0.05)	**1.00** (0.00)	0.90 (0.06)

δ-robustness VeriTraCER exhibits high levels of δ-robustness: 70.24% (±17.41%) of test samples exhibit δ-robustness for HELOC, 81.08% (±4.51%) for TC, and 96.96% (±0.96%) for OULA. For the same value of δ, CN has a δ-robustness for 26.29% (±18.29%) of HELOC samples, but 0% for TC and OULA samples. Likewise, the CEs generated by ROAR and SNS for models trained in a standard way have 0% δ-robustness on all three datasets.

Table 2: Fraction of samples whose CEs are valid and robust after finetuning with a distribution shift. Standard deviation over 10 trials in parentheses.

	CTG	WHO
VeriTraCER	**0.987** (0.010)	**0.995** (0.027)
CN	0.981 (0.019)	0.967 (0.030)
ROAR	0.570 (0.104)	0.884 (0.078)
SNS	0.407 (0.024)	0.846 (0.050)

Random initialization and leave-one-out results Table 1 shows what fraction of counterfactuals, generated for a particular model, remain valid for a model trained (a) using a different random seed, or (b) with a different 1% of the training data removed. We note that most CE and dataset combinations have high ($> 90\%$) cross-model validity rates; however, VeriTraCER has the highest or second-highest cross-model validity rate for most settings.

Real-world distribution shifts Table 2 shows the robustness of CEs to real-world distribution shifts. We see that VeriTraCER significantly outperforms the non-CounterNet baseline methods, especially on CTG.

Counterfactual Quality A weakness of VeriTraCER is that the generated CEs are a larger distance from the original CEs. The data on proximity (the l_1-distance between an input $\mathbf{x}$ and its CE $\mathbf{x}'$ is) is summarized in Table 3. Note that the two datasets where VeriTraCER performs worst relative to the other techniques (TC and OULA) have categorical features, which ROAR does not modify, and SNS does not handle properly (i.e., by breaking one-hot encodings).

Table 3: Average proximity for valid CEs. (*) indicates that the CE technique does not correctly account for categorical features.

	CTG	WHO	HELOC	TC	OULA
VeriTraCER	0.250	0.216	0.099	0.244	0.179
CN	0.220	0.164	0.107	0.234	0.169
ROAR	0.189	0.031	**0.032**	0.018*	0.015*
SNS	**0.038**	**0.031**	0.041	**0.015***	**0.013***

6 Conclusions

We have presented VeriTraCER, a training algorithm that jointly produces a model f and a CE generator g such that the CEs generated by g will be robust to small changes in the weights of f. We do this by minimizing an upper bound on our robust loss, i.e., we minimize the loss on the *worst-case* model in the multiplicity set $\mathcal{M}_{f,\mathbf{x}}$. We provide a refinement of interval-bound propagation, Simul-CROWN, allowing the over-approximation to be tighter than other state-of-the-art approaches. Our approach is able to to find CEs that are *certifiably robust* at high rates (typically over 90%). This carries over to high robustness for empirical model updates, such as retraining with a different random seed. In particular, we outperform state-of-the-art approaches on finetuning after real-world distribution shifts. The tradeoff is that VeriTraCER generally yields CEs that are a larger distance form the original sample. In some settings, the additional robustness as well as fast CE generation time may be worth the additional recourse costs, but future work should aim to provide similar deterministic guarantees and empirical performance with smaller CE distances.

REPRODUCIBILITY STATEMENT

Our code is available at `https://github.com/ForeverZyh/robust_cfx`.

REFERENCES

Christopher Anders, Plamen Pasliev, Ann-Kathrin Dombrowski, Klaus-Robert Müller, and Pan Kessel. Fairwashing explanations with off-manifold detergent. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 314–323. PMLR, 13–18 Jul 2020. URL `https://proceedings.mlr.press/v119/anders20a.html`.

Emily Black, Zifan Wang, Matt Fredrikson, and Anupam Datta. Consistent counterfactuals for deep models. *arXiv preprint arXiv:2110.03109*, 2021.

Ngoc Bui, Duy Nguyen, and Viet Anh Nguyen. Counterfactual Plans under Distributional Ambiguity. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=noaG7SrPVK0`.

D. Campos and J. Bernardes. Cardiotocography. UCI Machine Learning Repository, 2010. DOI: https://doi.org/10.24432/C51S4N.

Consumer Financial Protection Bureau. 12 CFR Part 1002 - Equal Credit Opportunity Act (Regulation B). Technical report, Consumer Financial Protection Bureau, January 2018. URL `https://www.consumerfinance.gov/rules-policy/regulations/1002/`.

Patrick Cousot and Radhia Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the 4th ACM Symposium on Principles of Programming Languages, POPL 1977, Los Angeles, California, USA, January 1977*, pp. 238–252, 1977. doi: 10.1145/512950.512973. URL `https://doi.org/10.1145/512950.512973`.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D. Hoffman, Farhad Hormozdiari, Neil Houlsby, Shaobo Hou, Ghassen Jerfel, Alan Karthikesalingam, Mario Lucic, Yian Ma, Cory McLean, Diana Mincu, Akinori Mitani, Andrea Montanari, Zachary Nado, Vivek Natarajan, Christopher Nielson, Thomas F. Osborne, Rajiv Raman, Kim Ramasamy, Rory Sayres, Jessica Schrouff, Martin Seneviratne, Shannon Sequeira, Harini Suresh, Victor Veitch, Max Vladymyrov, Xuezhi Wang, Kellie Webster, Steve Yadlowsky, Taedong Yun, Xiaohua Zhai, and D. Sculley. Underspecification presents challenges for credibility in modern machine learning. *J. Mach. Learn. Res.*, 23(1), jan 2022. ISSN 1532-4435.

Ann-Kathrin Dombrowski, Christopher J Anders, Klaus-Robert Müller, and Pan Kessel. Towards robust explanations for deep neural networks. *Pattern Recognition*, 121:108194, 2022.

Sanghamitra Dutta, Jason Long, Saumitra Mishra, Cecilia Tilli, and Daniele Magazzeni. Robust Counterfactual Explanations for Tree-Based Ensembles. In *International Conference on Machine Learning*, pp. 5742–5756. PMLR, 2022.

European Commission. General Data Protection Regulation (GDPR), 2018. URL `https://gdpr.eu/tag/gdpr/`.

FICO. Explainable Machine Learning Challenge, 2018. URL `https://community.fico.com/s/explainable-machine-learning-challenge?tabset-158d9=d157e`.

Alexandre Forel, Axel Parmentier, and Thibaut Vidal. Robust Counterfactual Explanations for Random Forests. *arXiv preprint arXiv:2205.14116*, 2022.

Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy A. Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *CoRR*, abs/1810.12715, 2018. URL http://arxiv.org/abs/1810.12715.

Hangzhi Guo, Thanh Hong Nguyen, and Amulya Yadav. Counternet: End-to-end training of prediction aware counterfactual explanations. In Ambuj Singh, Yizhou Sun, Leman Akoglu, Dimitrios Gunopulos, Xifeng Yan, Ravi Kumar, Fatma Ozcan, and Jieping Ye (eds.), *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, pp. 577–589. ACM, 2023. doi: 10.1145/3580305.3599290. URL https://doi.org/10.1145/3580305.3599290.

Faisal Hamman, Erfaun Noorani, Saumitra Mishra, Daniele Magazzeni, and Sanghamitra Dutta. Robust counterfactual explanations for neural networks with probabilistic guarantees. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 12351–12367. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/hamman23a.html.

Juyeon Heo, Sunghwan Joo, and Taesup Moon. *Fooling Neural Network Interpretations via Adversarial Model Manipulation*. Curran Associates Inc., Red Hook, NY, USA, 2019.

Junqi Jiang, Francesco Leofante, Antonio Rago, and Francesca Toni. Formalising the robustness of counterfactual explanations for neural networks. In Brian Williams, Yiling Chen, and Jennifer Neville (eds.), *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pp. 14901–14909. AAAI Press, 2023. doi: 10.1609/aaai.v37i12.26740. URL https://doi.org/10.1609/aaai.v37i12.26740.

Amir-Hossein Karimi, Gilles Barthe, Borja Balle, and Isabel Valera. Model-agnostic counterfactual explanations for consequential decisions. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 895–905. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/karimi20a.html.

Jakub Kuzilek, Martin Hlosta, and Zdenek Zdrahal. Open university learning analytics dataset. *Scientific Data*, 4:170171, 2017. doi: 10.1038/sdata.2017.171.

Arnaud Van Looveren and Janis Klaise. Interpretable Counterfactual Explanations Guided by Prototypes. *ArXiv*, abs/1907.02584, 2019.

Scott M Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf.

Charles Marx, Flavio Calmon, and Berk Ustun. Predictive multiplicity in classification. In *International Conference on Machine Learning*, pp. 6765–6774. PMLR, 2020.

Anna P. Meyer, Dan Ley, Suraj Srinivas, and Himabindu Lakkaraju. On Minimizing the Impact of Dataset Shifts on Actionable Explanations. In *The 39th Conference on Uncertainty in Artificial Intelligence*, 2023. URL https://openreview.net/forum?id=mtd904kJUs.

Duy Nguyen, Ngoc Bui, and Viet Anh Nguyen. Distributionally robust recourse action. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=E3ip6qBLF7.

Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. On counterfactual explanations under predictive multiplicity. In *Conference on Uncertainty in Artificial Intelligence*, pp. 809–818. PMLR, 2020.

Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. FACE: Feasible and Actionable Counterfactual Explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, AIES '20, pp. 344–350, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450371100. doi: 10.1145/3375627.3375850. URL `https://doi.org/10.1145/3375627.3375850`.

Kumar Rajarshi. Life Expectancy (WHO), 2017. URL `https://www.kaggle.com/datasets/kumarajarshi/life-expectancy-who`.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi (eds.), *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pp. 1135–1144. ACM, 2016. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939778.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *CoRR*, abs/1312.6034, 2013.

Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin T. Vechev. An abstract domain for certifying neural networks. *Proc. ACM Program. Lang.*, 3(POPL):41:1–41:30, 2019. doi: 10.1145/3290354. URL `https://doi.org/10.1145/3290354`.

Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, AIES '20, pp. 180–186, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450371100. doi: 10.1145/3375627.3375830. URL `https://doi.org/10.1145/3375627.3375830`.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

Suraj Srinivas, Kyle Matoba, Himabindu Lakkaraju, and François Fleuret. Efficient Training of Low-Curvature Neural Networks. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=2B2xIJ299rx`.

Sohini Upadhyay, Shalmali Joshi, and Himabindu Lakkaraju. Towards robust and reliable algorithmic recourse. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 16926–16937, 2021. URL `https://proceedings.neurips.cc/paper/2021/hash/8ccfb1140664a5fa63177fb6e07352f0-Abstract.html`.

Berk Ustun, Alexander Spangher, and Yang Liu. Actionable recourse in linear classification. *arXiv preprint arXiv:1809.06514*, 2018.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual Explanations Without Opening the Black Box: Automated Decisions and the GDPR. *Harvard Journal of Law & Technology*, 31: 841–887, 04 2018. doi: 10.2139/ssrn.3063289.

I-Cheng Yeh and Che-hui Lien. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, 36(2, Part 1):2473–2480, 2009. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2007.12.020. URL `https://www.sciencedirect.com/science/article/pii/S0957417407006719`.

Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 4944–4953, 2018. URL `https://proceedings.neurips.cc/paper/2018/hash/d04863f100d59b3eb688a11f95b0ae60-Abstract.html`.

Huan Zhang, Hongge Chen, Chaowei Xiao, Sven Gowal, Robert Stanforth, Bo Li, Duane S. Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neural networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL `https://openreview.net/forum?id=Skxuk1rFwB`.

Yuhao Zhang, Aws Albarghouthi, and Loris D'Antoni. Certified robustness to programmable transformations in lstms. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pp. 1068–1083. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.EMNLP-MAIN.82. URL `https://doi.org/10.18653/v1/2021.emnlp-main.82`.

Algorithm 1 Joint Training of f and g

```
Input: Training set $D = (\mathbf{x}_i, y_i)_{i=1}^n$, Hyper-parameters $\delta, p$ in Definition 3.1
Output: Classifier $f$ and robust CE generator $g$
1: Initialize $f$ and $g$
2: for $e = 1$ to maxepoch do
3:     $\mathbf{x}' = g(f, \mathbf{x}_1), g(f, \mathbf{x}_2), \ldots, g(f, \mathbf{x}_n)$
4:     Optimize $\theta_f$ via $\nabla_{\theta_f} \sum_{i=1}^n \mathcal{L}_f(\mathbf{x}_i, y_i, \mathbf{x}'_i, \theta_f, \delta, p)$
5:     Optimize $\theta_g$ via $\nabla_{\theta_g} \sum_{i=1}^n \mathcal{L}_g(\mathbf{x}_i, y_i, g(f, \mathbf{x}_i), \theta_f, \delta, p)$
6: end for
7: return $f, g$
```

Algorithm 2 Solving Equation (3) when $\hat{y} = 1$

```
Input: Bounded parameters $\theta^\sharp = (\theta^L, \theta^U)$, and coefficients from CROWN-IBP $\boldsymbol{\mu}^u, \nu^u, \boldsymbol{\alpha}^u, \beta^u$
Output: $\max_{\theta \in \theta^\sharp} \boldsymbol{\mu}^u \theta + \nu^u$   s.t. $\boldsymbol{\alpha}^u \theta + \beta^u \geq 0$
for $i = 1$ to $n$ do
    if $\boldsymbol{\alpha}_i^u > 0 \vee (\boldsymbol{\alpha}_i^u = 0 \wedge \boldsymbol{\mu}_i^u > 0)$ then
        $\theta_i = \theta_i^U$
    else
        $\theta_i = \theta_i^L$
    end if
end for
$s \leftarrow \boldsymbol{\alpha}^u \theta + \beta^u$, $s' \leftarrow \boldsymbol{\mu}^u \theta + \nu^u$
if $s < 0$ then
    return $-\infty$                                        // Constraint not satisfied
end if
$I \leftarrow \{1 \leq i \leq n \mid \boldsymbol{\mu}_i^u \boldsymbol{\alpha}_i^u < 0\}$
Sort the indices list $I$ descendingly by $-\frac{\boldsymbol{\mu}_i^u}{\boldsymbol{\alpha}_i^u}$
for $i \in I$ do
    $\delta \leftarrow |\boldsymbol{\alpha}_i^u|(\theta^U - \theta^L)$, $\delta' \leftarrow |\boldsymbol{\mu}_i^u|(\theta^U - \theta^L)$
    if $\delta > s$ then
        $s' \leftarrow s' + \delta' \frac{s}{\delta}$
        break
    end if
    $s \leftarrow s - \delta$, $s' \leftarrow s' + \delta'$
end for
return $s'$
```

A Algorithms

Algorithm 1 gives an overview of our robust training approach. We concurrently train a model f and a CE generator g. Alternatively, it is possible to use a fixed CE generator and to train f to be a model whose CEs – with respect to the fixed g – are likely to be robust. In that case, line 1 of the algorithm only initializes f, and line 5 of the algorithm is skipped.

Algorithm 2 shows our algorithm to solve Equation (3). Notably, this algorithm – when implemented in PyTorch – preserves gradient information. If we were to naïvely solve a linear programming (LP) problem to optimize the RobustCE loss within each training batch, the gradient information would be lost and thus the optimization process would not work.

B Proofs

We provide proofs of Theorems 4.1 and 4.3.

Proof of Theorem 4.1. First, we will show that the IBP loss provides a sound upper bound on $\mathcal{L}_{\mathrm{R}}$. By the soundness of IBP, we have

$$\mathcal{L}_{\mathrm{R}}^{\sharp\mathrm{IBP}}(\mathbf{x}, \mathbf{x}', \theta_f) \geq \max_{f_{\mathrm{m}} \in \mathcal{M}_f} \mathcal{L}_{\mathrm{MSE}}(\bar{f}_{\mathrm{m}}(\mathbf{x}'), 1 - \hat{y})$$

Table 4: Dataset composition

Dataset	Size (orig.)	Size (shifted)	# cont. feat.	# cat. feat.
HELOC	9871	-	22	0
TC	30000	-	14	9
OULA	32593	-	22	8
CTG	1950	2126	22	0
WHO	2196	2928	18	0

Next, note that $\mathcal{M}_{f,\mathbf{x}} \subseteq \mathcal{M}_f$ ensures the overapproximation can still capture the worst-case classifier. Formally,

$$\max_{f_{\rm m} \in \mathcal{M}_f} \mathcal{L}_{\rm MSE}(\bar{f_{\rm m}}(\mathbf{x}'), 1-\hat{y}) \geq \mathcal{L}_{\rm R}(\mathbf{x}, \mathbf{x}', \theta_f)$$

The proof of CROWN-IBP loss is tighter than the IBP loss is given in Zhang et al. (2020). □

Proof of Theorem 4.3. Without loss of generality, we consider the case $\hat{y} = 1$. Simul-CROWN applies CROWN-IBP to obtain lower and upper bounds in $\mathcal{M}_f$. For all parameters $\theta \in \theta^\sharp$, we have $\sigma(\boldsymbol{\alpha}^l\theta + \beta^l) \leq \bar{f}_\theta(\mathbf{x}) \leq \sigma(\boldsymbol{\alpha}^u\theta + \beta^u)$ and $\sigma(\boldsymbol{\mu}^l\theta + \nu^l) \leq \bar{f}_\theta(\mathbf{x}) \leq \sigma(\boldsymbol{\mu}^u\theta + \nu^u)$. The coefficients $\boldsymbol{\alpha}^l, \beta^l, \boldsymbol{\alpha}^u, \beta^u, \boldsymbol{\mu}^l, \nu^l, \boldsymbol{\mu}^u, \nu^u$ in Simul-CROWN are the same as the ones in CROWN-IBP.

Simul-CROWN computes the upper bound of $\mathcal{L}_{\rm R}$ as $\mathcal{L}_{\rm R}^{\sharp{\rm Ours}}(\mathbf{x}, \mathbf{x}', \theta_f) = \mathcal{L}_{\rm MSE}(\sigma(t_{\textit{Ours}}), 0)$, where $t_{\textit{Ours}}$ is obtained in the following optimization problem,

$$t_{\textit{Ours}} = \max_{\theta \in \theta^\sharp} \boldsymbol{\mu}^u\theta + \nu^u \quad \text{s.t. } \boldsymbol{\alpha}^u\theta + \beta^u \geq 0. \tag{4}$$

CROWN-IBP computes the upper bound of $\mathcal{L}_{\rm R}$ as $\mathcal{L}_{\rm R}^{\sharp{\rm CIBP}}(\mathbf{x}, \mathbf{x}', \theta_f) = \mathcal{L}_{\rm MSE}(\sigma(t_{\textit{CIBP}}), 0)$, where $t_{\textit{CIBP}}$ is obtained in the following optimization problem,

$$t_{\textit{CIBP}} = \max_{\theta \in \theta^\sharp} \boldsymbol{\mu}^u\theta + \nu^u. \tag{5}$$

As Equation (4) has an additional constraint than Equation (5), we have $t_{\textit{CIBP}} \geq t_{\textit{Ours}}$. Therefore, we have $\mathcal{L}_{\rm MSE}(\sigma(t_{\textit{CIBP}}), 0) \geq \mathcal{L}_{\rm MSE}(\sigma(t_{\textit{Ours}}), 0)$, which leads to

$$\mathcal{L}_{\rm R}^{\sharp{\rm CIBP}}(\mathbf{x}, \mathbf{x}', \theta_f) \geq \mathcal{L}_{\rm R}^{\sharp{\rm Ours}}(\mathbf{x}, \mathbf{x}', \theta_f).$$

□

C Additional experimental results

C.1 Experimental setup

Details on datasets Details on dataset size and feature composition (number of features, as well as continuous/categorical breakdown) are summarized in Table 4. We create our own distribution shift for CTG and WHO as follows. CTG originally has three classes (normal, suspect, and pathological). We create a distribution shift by first training on only the “normal” and “suspect” samples. Then, we add in the ”pathological” samples (but assign them the same label as the “suspect” samples). WHO contains data about countries’ life expectancies across several years (2000-2015). We create a temporal data shift by using pre-2012 data as the original dataset, and all data as the shifted dataset. When preprocessing WHO, we also binarize the outcome variable. Instead of predicting the exact life expectancy in years, we predict whether or not it is greater than the pre-2012 median life expectancy.

Choosing δ We dynamically determine a distinct δ_i value for the i-th layer with parameter θ_i based on its l_p norm, denoted as $\delta_i = \kappa\|\theta_i\|_p$, where $\kappa \in [0, 1]$ is the ratio of δ_i to the layer norm. This design is motivated by two considerations. Firstly, parameters across different layers have different l_p norms due to layer size and depth differences. Consequently, a fixed δ may be overly restrictive for some layers and lenient for others. Secondly, using a single fixed δ could potentially inflate the norm of each layer artificially to enhance δ-Robustness because larger layer norms make the δ-robust multiplicity set relatively smaller under a fixed δ.

C.2 Class-level data on CTG

We include additional analysis on class-level performance for CTG. Recall that the original dataset has 3 classes: normal (N), suspect (S), and pathological (P). We train the original model only on N and S data, and then finetune a shifted model with the addition of P data (assigned to the same class as S). We perform this analysis because the dataset is very imbalanced: around 78% belongs to class N. That is, we want to ensure that the finetuning process actually adapts the model and CE generator to the shifted data (rather than just yielding an overall high performance at the expense of the minority groups S and P).

Before finetuning, all CEs are valid regardless of class. After finetuning, we find that overall, 98.7% of CEs generated for CTG remain valid. For samples with class=S (i.e., the minority samples that are represented in the training data), 95.9% of CEs are valid after finetuning. And for samples with class=P (i.e., only represented in the finetuning data), 96.8% of CEs remain valid. So, while robustness to finetuning is slightly lower for the minority classes, robustness is very high for all groups.

C.3 Ablation Studies

Other distance metrics In addition to proximity (see Section 5.1 for a definition), we consider *sparsity* and *distance to the data manifold (DDM)* as CE quality metrics. Sparsity is defined as the fraction of features that change, i.e., given $\mathbf{x}$ and its $\mathbf{x}'$, $\text{SPARS}(\mathbf{x}, \mathbf{x}') = \frac{1}{d}\sum_i^d \mathbb{1}[\mathbf{x}_i \neq \mathbf{x}'_i]$. We approximate DDM by taking the distance to the nearest point in the training data (by using k-nearest neighbors with $k = 1$). For all three metrics, a lower score indicates smaller distance, which indicates that the CE is of higher quality because it will be easier for the user to obtain. Table 5 contains data on sparsity and DDM; see Table 3 in the main paper for proximity data.

Table 5: Average sparsity and distance to the data manifold (DDM) for valid CE. (*) indicates that the CE technique does not correctly account for categorical features.

		CTG	WHO	HELOC	TC	OULA
Proximity	VeriTraCER	0.780	0.758	0.817	0.758	0.424
	CN	0.765	0.718	0.791	0.751	0.403
	ROAR	0.900	0.326	0.286	0.166*	0.399*
	SNS	0.585	0.338	0.541	0.210*	0.292*
DDM	VeriTraCER	0.130	0.103	0.050	0.123	0.051
	CN	0.109	0.064	0.049	0.108	0.039
	ROAR	0.072	0.042	0.048	0.020*	0.028*
	SNS	0.052	0.040	0.054	0.015*	0.024*

Relation of our approach's effectiveness and the tightness of the bound. We compare the three different techniques from Section 4.2. Using an looser bound such as IBP or CROWN-IBP in place of Simul-CROWN is advantageous for computation speed, however, the overly loose bounds may hurt the effectiveness. For instance, we expect to see lower model accuracy and higher CE distance metrics with IBP and CROWN-IBP than with Simul-CROWN. However, Table 6 shows that three techniques perform largely similarly. Note that the δ-Robustness rate is computed post-training using Simul-CROWN as it achieves tighter over-approximation than the other two approaches.

C.4 Time complexity

For VeriTraCER, it takes 8.24 (±0.14) s. to train one epoch on HELOC, 126.61 (±1.92) s. to train one epoch on OULA, and 27.66 (±0.36) s. to train one epoch on TC. By contrast, training one epoch with standard training on the same hardware takes 0.55 (±0.01) s. for HELOC, 5.39 (±0.06) s. for OULA, and 1.80 (±0.01) s. for TC. We train each dataset for 100 epochs, so with VeriTraCER, it takes on the order of 14 minutes for HELOC, 3 hours for OULA, 45 minutes for TC. With standard training, those times are reduced to around 1 minute, 9 minutes, and 3 minutes, respectively.

Table 7 summarizes the time required to generate one CE, given a trained model. (CounterNet is not included, as it takes the same amount of time as VeriTraCER.) We see that VeriTraCER can generate

Table 6: Fraction of samples that exhibit δ-robustness and empirical random initialization (RI) robustness, along with test accuracy and CE quality metrics for IBP, CROWN-IBP, and VeriTraCER. Prox. is proxmity, Spars. is sparsity, and Man. is distance to the data manifold.

		Δ-Rob.	RI	Acc.	Prox.	Spars.	Man.
HELOC	IBP	0.746	0.920 (0.079)	0.742	0.10 (0.03)	0.82 (0.08)	0.05 (0.01)
	C-IBP	0.721	0.918 (0.064)	0.742	0.10 (0.03)	0.82 (0.08)	0.05 (0.01)
	VeriTraCER	0.788	0.932 (0.028)	0.738	0.10 (0.03)	0.82 (0.08)	0.05 (0.01)
TC	IBP	0.902	0.862 (0.187)	0.812	0.22 (0.06)	0.74 (0.09)	0.10 (0.03)
	C-IBP	0.872	0.900 (0.120)	0.813	0.22 (0.06)	0.75 (0.09)	0.10 (0.03)
	VeriTraCER	0.813	0.884 (0.141)	0.806	0.24 (0.05)	0.76 (0.09)	0.12 (0.03)
OULA	IBP	0.880	0.922 (0.079)	0.929	0.19 (0.04)	0.45 (0.12)	0.06 (0.02)
	C-IBP	0.968	0.941 (0.055)	0.930	0.18 (0.04)	0.43 (0.12)	0.05 (0.02)
	VeriTraCER	0.970	0.941 (0.054)	0.929	0.18 (0.04)	0.42 (0.12)	0.05 (0.02)

Table 7: Average time, in seconds, to generate one counterfactual given a trained model. The standard deviation for VeriTraCER is negligible ($< 10^{-4}$) for all datasets.

	HELOC	OULA	TC
VeriTraCER	<0.001	<0.001	<0.001
ROAR	3.25 ±0.72	2.75 ±0.52	3.33 ±0.40
SNS	5.25 ±0.21	5.92 ±0.21	5.89 ±0.20

CEs near-instantaneously, while ROAR and SNS must solve complex optimization problems for each instance.