
COUNTERFACTUAL TRAINING: TEACHING MODELS PLAUSIBLE AND ACTIONABLE EXPLANATIONS

A PREPRINT

Patrick Altmeyer 

Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

p.altmeyer@tudelft.nl

Aleksander Buszydlik

Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Arie van Deursen

Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Cynthia C. S. Liem

Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

February 24, 2025

ABSTRACT

Counterfactual Explanations have emerged as a popular tool to explain predictions made by opaque machine learning models: they explain how factual inputs need to change in order for some fitted model to produce some desired output. Much existing research has focused on identifying explanations that are not only valid but also deemed plausible and desirable with respect to the underlying data and stakeholder requirements. Recent work has shown that under this premise, the task of learning plausible explanations is effectively reassigned from the model itself to the (post-hoc) counterfactual explainer. Building on that work, we propose a novel model objective that leverages counterfactuals during the training phase (ad-hoc) in order to minimize the divergence between learned representations and plausible explanations. Through extensive experiments, we demonstrate that our proposed methodology facilitates training models that inherently deliver plausible explanations while maintaining high predictive performance.

Keywords Counterfactual Explanations • Explainable AI • Representation Learning

1 Introduction

Today's prominence of artificial intelligence (AI) has largely been driven by advances in **representation learning**: instead of relying on features and rules that are carefully hand-crafted by humans, modern machine learning (ML) models are tasked with learning these representations from scratch, guided by narrow objectives such as predictive accuracy ([I. Goodfellow, Bengio, and Courville 2016](#)). Modern advances in computing have made it possible to

18 provide such models with ever greater degrees of freedom to achieve that task, which has often led them to outperform
 19 traditionally more parsimonious models. Unfortunately, in doing so they also learn increasingly complex and highly
 20 sensitive representations that we can no longer easily interpret.

21 This trend towards complexity for the sake of performance has come under serious scrutiny in recent years. At the very
 22 cusp of the deep learning revolution, Szegedy et al. (2013) showed that artificial neural networks (ANN) are sensitive
 23 to adversarial examples: counterfactuals of model inputs that yield vastly different model predictions despite being
 24 “imperceptible” in that they are semantically indifferent from their factual counterparts. Despite partially effective
 25 mitigation strategies such as **adversarial training** (I. J. Goodfellow, Shlens, and Szegedy 2014), truly robust deep
 26 learning (DL) remains unattainable even for models that are considered shallow by today’s standards (Kolter 2023).

27 Part of the problem is that high degrees of freedom provide room for many solutions that are locally optimal with
 28 respect to narrow objectives (Wilson 2020)¹. Based purely on predictive performance, these solutions may seem to
 29 provide compelling explanations for the data, when in fact they are based on purely associative, semantically mean-
 30 ingless patterns. This poses two related challenges: firstly, it makes these models inherently opaque, since humans
 31 cannot simply interpret what type of explanation the complex learned representations correspond to; secondly, even
 32 if we could resolve the first challenge, it is not obvious how to mitigate models from learning representations that
 33 correspond to meaningless and implausible explanations.

34 The first challenge has attracted an abundance of research on **explainable AI** (XAI) which aims to develop tools to
 35 derive explanations from complex model representations. This can mitigate a scenario in which we deploy opaque
 36 models and blindly rely on their predictions. On countless occasions, this scenario has already occurred in practice
 37 and caused real harm to people who were affected adversely and often unfairly by automated decision-making systems
 38 (ADMS) involving opaque models (O’Neil 2016). Effective XAI tools can aide us in monitoring models and providing
 39 recourse to individuals to turn adverse outcomes (e.g. “loan application rejected”) into positive ones (“application
 40 accepted”). Wachter, Mittelstadt, and Russell (2017) propose **counterfactual explanations** as an effective approach
 41 to achieve this: they explain how factual inputs need to change in order for some fitted model to produce some desired
 42 output, typically involving minimal perturbations.

43 To our surprise, the second challenge has not yet attracted any consolidated research effort. Specifically, there has
 44 been no concerted effort towards improving model **explainability**, which we define here as the degree to which learned
 45 representations correspond to explanations that are interpretable and deemed **plausible** by humans (see Definition 3.1).
 46 Instead, the choice has typically been to improve the capacity of XAI tools to identify the subset explanations that are
 47 both plausible and valid for any given model, independent of whether the learned representations are also compatible
 48 with implausible explanations (Altmeyer et al. 2024). Fortunately, recent findings indicate that explainability can arise
 49 as byproduct of regularization techniques aimed at other objectives such as robustness, generalization and generative
 50 capacity Altmeyer et al. (2024).

51 Building on these findings, we introduce **counterfactual training**: a novel regularization technique geared explicitly
 52 towards aligning model representations with plausible explanations. Our contributions are as follows:

- 53 • We discuss existing related work on improving models and consolidate it through the lens of counterfactual
 54 explanations (Section 2).
- 55 • We present our proposed methodological framework that leverages faithful counterfactual explanations during
 56 the training phase of models to achieve the explainability objective (Section 3).
- 57 • Through extensive experiments we demonstrate the counterfactual training improve model explainability
 58 while maintaining high predictive performance. We run ablation studies and grid searches to understand
 59 how the underlying model components and hyperparameters affect outcomes. (Section 4).

60 Despite limitations of our approach discussed in Section 5, we conclude that counterfactual training provides a practi-
 61 cal framework for researchers and practitioners interested in making opaque models more trustworthy Section 6. We
 62 also believe that this work serves as an opportunity for XAI researchers to reevaluate the premise of improving XAI
 63 tools without improving models.

64 2 Related Literature

65 To the best of our knowledge, our proposed framework for counterfactual training represents the first attempt to use
 66 counterfactual explanations during training to improve model explainability. In high-level terms, we define model
 67 explainability as the extent to which valid explanations derived for an opaque model are also deemed plausible with
 68 respect to the underlying data and stakeholder requirements. To make this more concrete, we follow Augustin, Meinke,

¹For clarity: we follow standard ML convention in using “degrees of freedom” to refer to the number of parameters estimated from data.

69 and Hein (2020) in tying the concept of explainability to the quality of counterfactual explanations that we can
 70 generate for a given model. The authors show that counterfactual explanations—understood here as minimal input
 71 perturbations that yield some desired model prediction—are generally more meaningful if the underlying model is
 72 more robust to adversarial examples. We can make intuitive sense of this finding when looking at adversarial training
 73 (AT) through the lens of representation learning with high degrees of freedom: by inducing models to “unlearn”
 74 representations that are susceptible to worst-case counterfactuals (i.e. adversarial examples), AT effectively removes
 75 some implausible explanations from the solution space.

76 2.1 Adversarial Examples are Counterfactual Explanations

77 This interpretation of the link between explainability through counterfactuals on one side, and robustness to adversarial
 78 examples on the other, is backed by empirical evidence. Sauer and Geiger (2021) demonstrate that using counterfactual
 79 images during classifier training improves model robustness. Similarly, Abbasnejad et al. (2020) argue that counter-
 80 factuals represent potentially useful training data in machine learning, especially in supervised settings where inputs
 81 may be reasonably mapped to multiple outputs. They, too, demonstrate the augmenting the training data of image
 82 classifiers can improve generalization. Teney, Abbasnejad, and Hengel (2020) propose an approach using counterfac-
 83 tuals in training that does not rely on data augmentation: they argue that counterfactual pairs typically already exist in
 84 training datasets. Specifically, their approach relies on, firstly, identifying similar input samples with different annota-
 85 tions and, secondly, ensuring that the gradient of the classifier aligns with the vector between pairs of counterfactual
 86 inputs using the cosine distance as a loss function.

87 In the natural language processing (NLP) domain, counterfactuals have similarly been used to improve models through
 88 data augmentation: Wu et al. (2021), propose *Polyjuice*, a general-purpose counterfactual generator for language mod-
 89 els. They demonstrate empirically that augmenting training data through *Polyjuice* counterfactuals improves robust-
 90 ness in a number of NLP tasks. Balashankar et al. (2023) also use *Polyjuice* to augment NLP datasets through diverse
 91 counterfactuals and show that classifier robustness improves up to 20%. Finally, Luu and Inoue (2023) introduce
 92 Counterfactual Adversarial Training (CAT), which also aims at improving generalization and robustness of language
 93 models. Specifically, they propose to proceed as follows: firstly, they identify training samples that are subject to
 94 high predictive uncertainty; secondly, they generate counterfactual explanations for those samples; and, finally, they
 95 fine-tune the given language model on the augmented dataset that includes the generated counterfactuals.

96 There have also been several attempts at formalizing the relationship between counterfactual explanations (CE) and
 97 adversarial examples (AE). Pointing to clear similarities in how CE and AE are generated, Freiesleben (2022) makes
 98 the case for jointly studying the opaqueness and robustness problem in representation learning. Formally, AE can
 99 be seen as the subset of CE, for which misclassification is achieved (Freiesleben 2022). Similarly, Pawelczyk et
 100 al. (2022) show that CE and AE are equivalent under certain conditions and derive theoretical upper bounds on the
 101 distances between them.

102 Two recent works are closely related to ours in that they use counterfactuals during training with the explicit goal
 103 of affecting certain properties of post-hoc counterfactual explanations. Firstly, Ross, Lakkaraju, and Bastani (2024)
 104 propose a way to train models that are guaranteed to provide recourse for individuals to move from an adverse outcome
 105 to some positive target class with high probability. The approach proposed by Ross, Lakkaraju, and Bastani (2024)
 106 builds on adversarial training, where in this context susceptibility to targeted adversarial examples for the positive
 107 class is explicitly induced. The proposed method allows for imposing a set of actionability constraints ex-ante: for
 108 example, users can specify that certain features (e.g. *age*, *gender*, ...) are immutable. Secondly, Guo, Nguyen, and
 109 Yadav (2023) are the first to propose an end-to-end training pipeline that includes counterfactual explanations as part
 110 of the training procedure. In particular, they propose a specific network architecture that includes a predictor and CE
 111 generator network, where the parameters of the CE generator network are learnable. Counterfactuals are generated
 112 during each training iteration and fed back to the predictor network. In contrast to Guo, Nguyen, and Yadav (2023),
 113 we impose no restrictions on the neural network architecture at all.

114 2.2 Beyond Robustness

115 Improving the adversarial robustness of models is not the only path towards aligning representations with plausible
 116 explanations. In a work closely related to this one, Altmeyer et al. (2024) show that explainability can be improved
 117 through model averaging and refined model objectives. The authors propose a way to generate counterfactuals that
 118 are maximally **faithful** to the model in that they are consistent with what the model has learned about the underlying
 119 data. Formally, they rely on tools from energy-based modelling to minimize the divergence between the distribution
 120 of counterfactuals and the conditional posterior over inputs learned by the model. Their proposed counterfactual
 121 explainer, *ECCCo*, yields plausible explanations if and only if the underlying model has learned representations that
 122 align with them. They find that both deep ensembles (Lakshminarayanan, Pritzel, and Blundell 2017) and joint energy-
 123 based models (JEMs) (Grathwohl et al. 2020) tend to do well in this regard.

124 Once again it helps to look at these findings through the lens of representation learning with high degrees of freedom.
 125 Deep ensembles are approximate Bayesian model averages, which are most called for when models are underspecified
 126 by the available data (Wilson 2020). Averaging across solutions mitigates the aforementioned risk of relying on a
 127 single locally optimal representations that corresponds to semantically meaningless explanations for the data. Previous
 128 work by Schut et al. (2021) similarly found that generating plausible (“interpretable”) counterfactual explanations is
 129 almost trivial for deep ensembles that have also undergone adversarial training. The case for JEMs is even clearer:
 130 they involve a hybrid objective that induces both high predictive performance and generative capacity (Grathwohl et al.
 131 2020). This is closely related to the idea of aligning models with plausible explanations and has inspired our proposed
 132 counterfactual training objective, as we explain in Section 3.

133 3 Counterfactual Training

134 Counterfactual training combines ideas from adversarial training, energy-based modelling and counterfactuals expla-
 135 nations with the explicit objective of aligning representations with plausible explanations that comply with user re-
 136 quirements. In the context of CE, plausibility has broadly been defined as the degree to which counterfactuals comply
 137 with the underlying data generating process (Poyiadzi et al. 2020; Guidotti 2022; Altmeyer et al. 2024). Plausibility
 138 is a necessary but insufficient condition for using CE to provide algorithmic recourse (AR) to individuals affected by
 139 opaque models in practice. This is because for recourse recommendations to be **actionable**, they need to not only
 140 result in plausible counterfactuals but also be attainable. A plausible CE for a rejected 20-year-old loan applicant, for
 141 example, might reveal that their application would have been accepted, if only they were 20 years older. Ignoring all
 142 other features, this complies with the definition of plausibility if 40-year-old individuals are in fact more credit-worthy
 143 on average than young adults. But of course this CE does not qualify for providing actionable recourse to the applicant
 144 since *age* is not a mutable feature. For our intents and purposes, counterfactual training aims at improving model ex-
 145 plainability by aligning models with counterfactuals that meet both desiderata, plausibility and actionability. Formally,
 146 we define explainability as follows:

147 **Definition 3.1** (Model Explainability). Let $\mathbf{M}_\theta : \mathcal{X} \mapsto \mathcal{Y}$ denote a supervised classification model that maps from the
 148 D -dimensional input space \mathcal{X} to representations $\phi(\mathbf{x}; \theta)$ and finally to the K -dimensional output space \mathcal{Y} . Assume
 149 that for any given input-output pair $\{\mathbf{x}, \mathbf{y}\}_i$ there exists a counterfactual $\mathbf{x}' = \mathbf{x} + \Delta : \mathbf{M}_\theta(\mathbf{x}') = \mathbf{y}^+ \neq \mathbf{y} = \mathbf{M}_\theta(\mathbf{x})$
 150 where $\arg \max_y \mathbf{y}^+ = y^+$ and y^+ denotes the index of the target class.

151 We say that \mathbf{M}_θ is **explainable** to the extent that faithfully generated counterfactuals are plausible (i.e. consistent with
 152 the data) and actionable. Formally, we define these properties as follows:

- 153 1. (Plausibility) $\int^A p(\mathbf{x}'|\mathbf{y}^+) d\mathbf{x} \rightarrow 1$ where A is some small region around \mathbf{x}' .
- 154 2. (Actionability) Permutations Δ are subject to actionability constraints.

155 We consider counterfactuals as faithful to the extent that they are consistent with what the model has learned about the
 156 input data. Let $p_\theta(\mathbf{x}|\mathbf{y}^+)$ denote the conditional posterior over inputs, then formally:

- 157 3. (Faithfulness) $\int^A p_\theta(\mathbf{x}'|\mathbf{y}^+) d\mathbf{x} \rightarrow 1$ where A is defined as above.

158 The definitions of faithfulness and plausibility in Definition 3.1 are the same as in Altmeyer et al. (2024), with adapted
 159 notation. Actionability constraints in Definition 3.1 vary and depend on the context in which \mathbf{M}_θ is deployed. In this
 160 work, we focus on domain and mutability constraints for individual features x_d for $d = 1, \dots, D$. We limit ourselves
 161 to classification tasks for reasons discussed in Section 5.

162 3.1 Our Proposed Objective

163 Let \mathbf{x}'_t for $t = 0, \dots, T$ denote a counterfactual explanation generated through gradient descent over T iterations
 164 as initially proposed by Wachter, Mittelstadt, and Russell (2017). For our purposes, we let T vary and consider the
 165 counterfactual search as converged as soon as the predicted probability for the target class has reached a pre-determined
 166 threshold, τ : $\mathcal{S}(\mathbf{M}_\theta(\mathbf{x}'))[y^+] \geq \tau^2$.
 167 To train models with high explainability as defined in Definition 3.1, we propose to leverage counterfactuals in the
 168 following objective,

$$\min_{\theta} \text{yloss}(\mathbf{M}_\theta(\mathbf{x}), \mathbf{y}) + \lambda_{\text{div}} \text{div}(\mathbf{x}, \mathbf{x}'_T, y; \theta) + \lambda_{\text{adv}} \text{advloss}(\mathbf{M}_\theta(\mathbf{x}'_{t \leq T}), \mathbf{y}) \quad (1)$$

²For detailed background information on gradient-based counterfactual search and convergence see Section B.1.

169 where $y\text{loss}(\cdot)$ is any conventional classification loss that induces discriminative performance (e.g. cross-entropy).
 170 The two additional components in Equation 1 are explained in more detail below. For now, they can be sufficiently de-
 171 scribed as inducing explainability directly and indirectly by penalizing: 1) the contrastive divergence, $\text{div}(\cdot)$, between
 172 counterfactuals \mathbf{x}_T' and observed samples x and, 2) the adversarial loss, $\text{advloss}(\cdot)$, with respect to nascent counterfac-
 173 tuals $\mathbf{x}'_{t \leq T}$. The tradeoff between the different components can be governed by adjusting the strengths of the penalties
 174 λ_{div} and λ_{adv} .

175 3.1.1 Directly Inducing Explainability through Contrastive Divergence

176 Grathwohl et al. (2020) observe that any classifier can be re-interpreted as a joint energy-based model (JEM)
 177 that learns to discriminate output classes conditional on inputs and generate inputs. They show that JEMs can be
 178 trained to perform well at both tasks by directly maximizing the joint log-likelihood factorized as $\log p_\theta(\mathbf{x}, \mathbf{y}) =$
 179 $\log p_\theta(\mathbf{y}|\mathbf{x}) + \log p_\theta(\mathbf{x})$. The first factor can be optimized using conventional cross-entropy as in Equation 1. To
 180 optimize $\log p_\theta(\mathbf{x})$ Grathwohl et al. (2020) minimize the contrastive divergence between samples drawn from $p_\theta(\mathbf{x})$
 181 and training observations, i.e. samples from $p(\mathbf{x})$.

182 A key empirical finding in Altmeyer et al. (2024) was that JEMs tend to do well with respect to the plausibility objec-
 183 tive in Definition 3.1. If we consider samples drawn from $p_\theta(\mathbf{x})$ as counterfactuals, this is an expected finding, because
 184 the JEM objective effectively minimizes the divergence between the conditional posterior and $p(\mathbf{x}|y^+)$. To generate
 185 samples, Grathwohl et al. (2020) rely on Stochastic Gradient Langevin Dynamics (SGLD) using an uninformative
 186 prior for initialization. This is where we depart from their methodology: instead of generating samples through SGLD,
 187 we propose using counterfactual explainers to generate counterfactuals for observed training samples. Specifically, we
 188 have

$$\text{div}(\mathbf{x}, \mathbf{x}'_T, y; \theta) = \mathcal{E}_\theta(\mathbf{x}, y) - \mathcal{E}_\theta(\mathbf{x}'_T, y) \quad (2)$$

189 where $\mathcal{E}_\theta(\cdot)$ denotes the energy function. In particular, we set $\mathcal{E}_\theta(\mathbf{x}, y) = -\mathbf{M}_\theta(\mathbf{x})[y^+]$ where y^+ denotes the index of
 190 the target class. We generate samples \mathbf{x}'_T by first randomly sampling the target class $y^+ \sim p(y)$ and then generating
 191 a counterfactual explanation for that target over T iterations using a gradient-based counterfactual generator. This is
 192 similar to how conditional sampling is used to draw from $p_\theta(\mathbf{x})$ in Grathwohl et al. (2020).

193 Intuitively, the gradient of Equation 2 decreases the energy of observed training samples (positive samples) while at
 194 same time increasing the energy of counterfactuals (negative samples) (Du and Mordatch 2020). As the generated
 195 counterfactuals get more plausible (Definition 3.1) over the cause of training, these two opposing effects gradually
 196 balance each out (Lippe 2024).

197 The departure from SGLD allows us to tap into the vast repertoire of explainers that have been proposed in the literature
 198 to meet different desiderata. Typically, these methods facilitate the imposition of domain and mutability constraints,
 199 for example. In principle, any existing approach for generating counterfactual explanations is viable, so long as it does
 200 not violate the faithfulness condition. Like JEMs (Murphy 2022), counterfactual training can be considered as a form
 201 of contrastive representation learning.

202 3.1.2 Indirectly Inducing Explainability through Adversarial Robustness

203 Based on our analysis in Section 2, counterfactuals \mathbf{x}' can be repurposed as additional training samples (Luu and Inoue
 204 2023; Balashankar et al. 2023) or adversarial examples (Freiesleben 2022; Pawelczyk et al. 2022). This leaves some
 205 flexibility with respect to the exact choice for $\text{advloss}(\cdot)$ in Equation 1. An intuitive functional form to use, though
 206 likely not the only reasonable choice, is inspired by adversarial training:

$$\begin{aligned} \text{advloss}(\mathbf{M}_\theta(\mathbf{x}'_{t \leq T}), \mathbf{y}; \varepsilon) &= y\text{loss}(\mathbf{M}_\theta(\mathbf{x}'_{t_\varepsilon}), \mathbf{y}) \\ t_\varepsilon &= \max_t \{t : \|\Delta_t\|_\infty < \varepsilon\} \end{aligned} \quad (3)$$

207 Under this choice, we consider nascent counterfactuals $\mathbf{x}'_{t \leq T}$ as adversarial examples as long as the magnitude of the
 208 perturbation to any individual feature is at most ε . This is closely aligned with Szegedy et al. (2013), who define an
 209 adversarial attack as an “imperceptible non-random perturbation”. Thus, we choose to work with a different distinction
 210 between CE and AE than Freiesleben (2022), who considers misclassification as the key distinguishing feature of AE.
 211 One of the key observations in this work is that we can leverage counterfactual explanations during training and get
 212 adversarial examples, essentially for free.

213 3.2 Encoding Actionability Constraints

214 Many existing counterfactual explainers support domain and mutability constraints out-of-the-box. In fact, both types
 215 of constraints can be implemented for any counterfactual explainer that relies on gradient descent in the feature space

216 for optimization (Altmeyer, Deursen, et al. 2023). In this context, domain constraints can be imposed by simply
 217 projecting counterfactuals back to the specified domain, if the previous gradient step resulted in updated feature values
 218 that were out-of-domain. Mutability constraints can similarly be enforced by setting partial derivatives to zero to
 219 ensure that features are only mutated in the allowed direction, if at all.

220 Since actionability constraints are binding at test time, we should also impose them when generating \mathbf{x}' during each
 221 training iteration to align model representations with user requirements. Through their effect on \mathbf{x}' , both types of
 222 constraints influence model outcomes through Equation 2. Here it is crucial that we avoid penalizing implausibility
 223 that arises due to mutability constraints. For any mutability-constrained feature d this can be achieved by enforcing
 224 $\mathbf{x}[d] - \mathbf{x}'[d] := 0$ whenever perturbing $\mathbf{x}'[d]$ in the direction of $\mathbf{x}[d]$ would violate mutability constraints. Specifically,
 225 we set $\mathbf{x}[d] := \mathbf{x}'[d]$ if

- 226 1. Feature d is strictly immutable in practice.
 227 2. We have $\mathbf{x}[d] > \mathbf{x}'[d]$ but feature d can only be decreased in practice.
 228 3. We have $\mathbf{x}[d] < \mathbf{x}'[d]$ but feature d can only be increased in practice.

229 From a Bayesian perspective, setting $\mathbf{x}[d] := \mathbf{x}'[d]$ can be understood as assuming a point mass prior for $p(\mathbf{x})$ with
 230 respect to feature d . Intuitively, we think of this simply in terms ignoring implausibility costs with respect to immutable
 231 features, which effectively forces the model to instead seek plausibility with respect to the remaining features. This
 232 in turn results in lower overall sensitivity to immutable features, which we demonstrate empirically for different
 233 classifiers in Section 4. Under certain conditions, this results holds theoretically[For the proof, see the supplementary
 234 appendix.]:

235 **Proposition 3.1** (Protecting Immutable Features). *Let $f_\theta(\mathbf{x}) = \mathcal{S}(\mathbf{M}_\theta(\mathbf{x})) = \mathcal{S}(\Theta\mathbf{x})$ denote a linear classifier with
 236 softmax activation \mathcal{S} (i.e. multinomial logistic regression) where $y \in \{1, \dots, K\} = \mathcal{K}$ and $\mathbf{x} \in \mathbb{R}^D$. If we assume
 237 multivariate Gaussian class densities with common diagonal covariance matrix $\Sigma_k = \Sigma$ for all $k \in \mathcal{K}$, then protecting
 238 an immutable feature from the contrastive divergence penalty (Equation 2) will result in lower classifier sensitivity to
 239 that feature relative to the remaining features, provided that at least one of those is mutable and discriminative.*

240 It is worth highlighting that Proposition 3.1 assumes independence of features. This raises a valid concern about the
 241 effect of protecting immutable features in the presence of proxy features that remain unprotected. We discuss this
 242 limitation in Section 5.

243 3.3 Illustration

244 To better convey the intuition underlying our proposed method, we illustrate different model outcomes in Example 3.1.

245 **Example 3.1** (Prediction of Consumer Credit Default). Suppose we are interested in predicting the likelihood that
 246 loan applicants default on their credit. We have access to historical data on previous loan takers comprised of a binary
 247 outcome variable ($y \in \{1 = \text{default}, 2 = \text{no default}\}$) two input features: 1) the subjects' *age*, which we define as
 248 immutable, and 2) the subjects' existing level of *debt*, which we define as mutable.

249 We have simulated this scenario using synthetic data with independent features and Gaussian class-conditional densities
 250 in Figure 1. The four panels in Figure 1 show the outcomes for different training procedures using the same model
 251 architecture each time (a linear classifier). In each case, we show the linear decision boundary (green) and the training
 252 data colored according to their ground-truth label: orange points belong to the target class, $y^+ = 2$, blue points belong
 253 to the non-target class, $y^- = 1$. Stars indicate counterfactuals in the target class generated at test time using generic
 254 gradient descent until convergence.

255 In panel (a), we have trained our model conventionally, and we do not impose mutability constraints at test time. The
 256 generated counterfactuals are all valid, but not plausible: they are clearly distinguishable from the ground-truth data. In
 257 panel (b), we have trained our model with counterfactual training, once again not imposing mutability constraints at test
 258 time. We observe that the counterfactuals are clearly plausible, therefore meeting the first objective of Definition 3.1.

259 In panel (c), we have used conventional training again, this time imposing the mutability constraint on *age* at test
 260 time. Counterfactuals are valid but involve some substantial reductions in *debt* for some individuals, in particular
 261 very young applicants. By comparison, counterfactual paths are shorter on average in panel (d), where we have used
 262 counterfactual training and protected immutable features as described in Section 3.2. In particular, we observe that due
 263 to the classifier's lower sensitivity to *age*, recourse recommendations with respect to *debt* are much more homogenous,
 264 in that they do not disproportionately punish younger individuals. The counterfactuals are also plausible with respect
 265 to the mutable feature. Thus, we consider the model in panel (d) as the most explainable according to Definition 3.1.

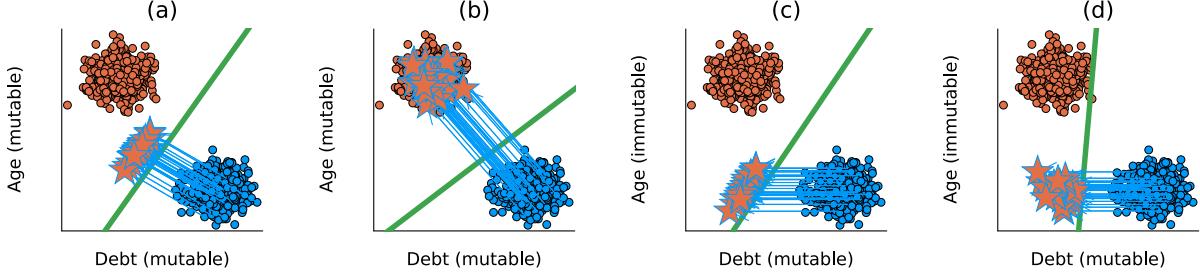


Figure 1: Visual illustration of how counterfactual training improves explainability. See Example 3.1 for details.

4 Experiments

In this section, we present experiments that we have conducted in order to answer the following research questions:

Research Question 4.1 (Plausibility). *Does our proposed counterfactual training objective (Equation 1) induce models to learn plausible explanations?*

Research Question 4.2 (Actionability). *Does our proposed counterfactual training objective (Equation 1) yield more favorable algorithmic recourse outcomes in the presence of actionability constraints?*

Beyond this, we are also interested in understanding how robust our answers to RQ 4.1 and RQ 4.2 are:

Research Question 4.3 (Hyperparameters). *What are the effects of different hyperparameter choices with respect to Equation 1?*

4.1 Experimental Setup

4.2 Experimental Results

5 Discussion

- 278 1. Limited to classification models.
- 279 2. Proxy attributes of immutable features.
- 280 3. Increased training time.
- 281 4. Training instabilities
- 282 5. Fairness and caveats (aware it's not a classical approach in this context, but there is a clear link).

6 Conclusion

References

- 285 Abbasnejad, Ehsan, Damien Teney, Amin Parvaneh, Javen Shi, and Anton van den Hengel. 2020. “Counterfactual
286 Vision and Language Learning.” In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition
287 (CVPR)*, 10041–51. <https://doi.org/10.1109/CVPR42600.2020.01006>.
- 288 Altmeyer, Patrick, Arie van Deursen, et al. 2023. “Explaining Black-Box Models Through Counterfactuals.” In
289 *Proceedings of the JuliaCon Conferences*, 1:130. 1.
- 290 Altmeyer, Patrick, Mojtaba Farmanbar, Arie van Deursen, and Cynthia CS Liem. 2024. “Faithful Model Explanations
291 Through Energy-Constrained Conformal Counterfactuals.” In *Proceedings of the AAAI Conference on Artificial
292 Intelligence*, 38:10829–37. 10.
- 293 Augustin, Maximilian, Alexander Meinke, and Matthias Hein. 2020. “Adversarial Robustness on in-and Out-
294 Distribution Improves Explainability.” In *European Conference on Computer Vision*, 228–45. Springer.
- 295 Balashankar, Ananth, Xuezhi Wang, Yao Qin, Ben Packer, Nithum Thain, Ed Chi, Jilin Chen, and Alex Beutel. 2023.
296 “Improving Classifier Robustness Through Active Generative Counterfactual Data Augmentation.” In *Findings of
297 the Association for Computational Linguistics: EMNLP 2023*, 127–39.
- 298 Bezanson, Jeff, Alan Edelman, Stefan Karpinski, and Viral B Shah. 2017. “Julia: A Fresh Approach to Numerical
299 Computing.” *SIAM Review* 59 (1): 65–98. <https://doi.org/10.1137/141000671>.
- 300 Bouchet-Valat, Milan, and Bogumi Kamiski. 2023. “DataFrames.jl: Flexible and Fast Tabular Data in Julia.” *Journal
301 of Statistical Software* 107 (4): 1–32. <https://doi.org/10.18637/jss.v107.i04>.
- 302 Chagas, Ronan Arraes Jardim, Ben Baumgold, Glen Hertz, Hendrik Ranocha, Mark Wells, Nathan Boyer, Nicholas
303 Ritchie, et al. 2024. “Ronisbr/PrettyTables.jl: V2.4.0.” Zenodo. <https://doi.org/10.5281/zenodo.13835553>.

- 304 Christ, Simon, Daniel Schwabedner, Christopher Rackauckas, Michael Krabbe Borregaard, and Thomas Breloff.
 305 2023. “Plots.jl – a User Extendable Plotting API for the Julia Programming Language.” <https://doi.org/https://doi.org/10.5334/joss.431>.
- 306 Danisch, Simon, and Julius Krumbiegel. 2021. “Makie.jl: Flexible High-Performance Data Visualization for Julia.”
 307 *Journal of Open Source Software* 6 (65): 3349. <https://doi.org/10.21105/joss.03349>.
- 308 Du, Yilun, and Igor Mordatch. 2020. “Implicit Generation and Generalization in Energy-Based Models.” <https://arxiv.org/abs/1903.08689>.
- 309 Freiesleben, Timo. 2022. “The Intriguing Relation Between Counterfactual Explanations and Adversarial Examples.”
 310 *Minds and Machines* 32 (1): 77–109.
- 311 Goodfellow, Ian J, Jonathon Shlens, and Christian Szegedy. 2014. “Explaining and Harnessing Adversarial Examples.”
 312 <https://arxiv.org/abs/1412.6572>.
- 313 Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.
- 314 Grathwohl, Will, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swer-
 315 sky. 2020. “Your Classifier Is Secretly an Energy Based Model and You Should Treat It Like One.” In *International
 316 Conference on Learning Representations*.
- 317 Guidotti, Riccardo. 2022. “Counterfactual Explanations and How to Find Them: Literature Review and Benchmark-
 318 ing.” *Data Mining and Knowledge Discovery*, 1–55.
- 319 Guo, Hangzhi, Thanh H. Nguyen, and Amulya Yadav. 2023. “CounterNet: End-to-End Training of Prediction Aware
 320 Counterfactual Explanations.” In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery
 321 and Data Mining*, 577–89. KDD ’23. New York, NY, USA: Association for Computing Machinery. <https://doi.org/10.1145/3580305.3599290>.
- 322 Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning*. Springer New
 323 York. <https://doi.org/10.1007/978-0-387-84858-7>.
- 324 Innes, Michael, Elliot Saba, Keno Fischer, Dhairyा Gandhi, Marco Conchetto Rudilosso, Neethu Mariya Joy, Tejan
 325 Karmali, Avik Pal, and Viral Shah. 2018. “Fashionable Modelling with Flux.” <https://arxiv.org/abs/1811.01457>.
- 326 Innes, Mike. 2018. “Flux: Elegant Machine Learning with Julia.” *Journal of Open Source Software* 3 (25): 602.
 327 <https://doi.org/10.21105/joss.00602>.
- 328 Kolter, Zico. 2023. “Keynote Addresses: SaTML 2023 .” In *2023 IEEE Conference on Secure and Trustworthy
 329 Machine Learning (SaTML)*, xvi–. Los Alamitos, CA, USA: IEEE Computer Society. <https://doi.org/10.1109/SaTML54575.2023.00009>.
- 330 Lakshminarayanan, Balaji, Alexander Pritzel, and Charles Blundell. 2017. “Simple and Scalable Predictive Uncer-
 331 tainty Estimation Using Deep Ensembles.” *Advances in Neural Information Processing Systems* 30.
- 332 Lippe, Phillip. 2024. “UvA Deep Learning Tutorials.” <https://uvadlc-notebooks.readthedocs.io/en/latest/>.
- 333 Luu, Hoai Linh, and Naoya Inoue. 2023. “Counterfactual Adversarial Training for Improving Robustness of Pre-
 334 Trained Language Models.” In *Proceedings of the 37th Pacific Asia Conference on Language, Information and
 335 Computation*, 881–88.
- 336 Murphy, Kevin P. 2022. *Probabilistic Machine Learning: An Introduction*. MIT Press.
- 337 O’Neil, Cathy. 2016. *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*.
 338 Crown.
- 339 Pawelczyk, Martin, Chirag Agarwal, Shalmali Joshi, Sohini Upadhyay, and Himabindu Lakkaraju. 2022. “Exploring
 340 Counterfactual Explanations Through the Lens of Adversarial Examples: A Theoretical and Empirical Analysis.”
 341 In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, edited by Gustau
 342 Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, 151:4574–94. Proceedings of Machine Learning Research.
 343 PMLR. <https://proceedings.mlr.press/v151/pawelczyk22a.html>.
- 344 Poyiadzi, Rafael, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. 2020. “FACE: Feasible and
 345 Actionable Counterfactual Explanations.” In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*,
 346 344–50.
- 347 Ross, Alexis, Himabindu Lakkaraju, and Osbert Bastani. 2024. “Learning Models for Actionable Recourse.” In
 348 *Proceedings of the 35th International Conference on Neural Information Processing Systems*. NIPS ’21. Red
 349 Hook, NY, USA: Curran Associates Inc.
- 350 Sauer, Axel, and Andreas Geiger. 2021. “Counterfactual Generative Networks.” <https://arxiv.org/abs/2101.06046>.
- 351 Schut, Lisa, Oscar Key, Rory Mc Grath, Luca Costabello, Bogdan Sacaleanu, Yarin Gal, et al. 2021. “Generating
 352 Interpretable Counterfactual Explanations By Implicit Minimisation of Epistemic and Aleatoric Uncertainties.” In
 353 *International Conference on Artificial Intelligence and Statistics*, 1756–64. PMLR.
- 354 Szegedy, Christian, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus.
 355 2013. “Intriguing Properties of Neural Networks.” <https://arxiv.org/abs/1312.6199>.
- 356 Teney, Damien, Ehsan Abbasnejad, and Anton van den Hengel. 2020. “Learning What Makes a Difference from
 357 Counterfactual Examples and Gradient Supervision.” In *Computer Vision–ECCV 2020: 16th European Conference,
 358 Glasgow, UK, August 23–28, 2020, Proceedings, Part x 16*, 580–99. Springer.
- 359

- 363 Wachter, Sandra, Brent Mittelstadt, and Chris Russell. 2017. “Counterfactual Explanations Without Opening the Black
364 Box: Automated Decisions and the GDPR.” *Harv. JL & Tech.* 31: 841. <https://doi.org/10.2139/ssrn.3063289>.
- 365 Wilson, Andrew Gordon. 2020. “The Case for Bayesian Deep Learning.” <https://arxiv.org/abs/2001.10995>.
- 366 Wu, Tongshuang, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel Weld. 2021. “Polyjuice: Generating Counterfactuals
367 for Explaining, Evaluating, and Improving Models.” In *Proceedings of the 59th Annual Meeting of the Association
368 for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing
369 (Volume 1: Long Papers)*, edited by Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, 6707–23. Online:
370 Association for Computational Linguistics. <https://doi.org/10.18653/v1/2021.acl-long.523>.

371 **A Notation**

- 372 • y^+ : The target class and also the index of the target class.
- 373 • y^- : The non-target class and also the index of non-the target class.
- 374 • \mathbf{y}^+ : The one-hot encoded output vector for the target class.
- 375 • θ : Model parameters (unspecified).
- 376 • Θ : Matrix of parameters.

377 **B Technical Details of Our Approach**

378 **B.1 Generating Counterfactuals through Gradient Descent**

379 In this section, we provide some background on gradient-based counterfactual generators (Section B.1.1) and discuss
380 how we define convergence in this context (Section B.1.2).

381 **B.1.1 Background**

382 **B.1.2 Convergence**

383 An important consideration when generating counterfactual explanations using gradient-based methods is how to
384 define convergence. Two common choices are to 1) perform gradient descent over a fixed number of iterations T , or
385 2) conclude the search as soon as the predicted probability for the target class has reached a pre-determined threshold,
386 $\tau: \mathcal{S}(\mathbf{M}_\theta(\mathbf{x}'))[y^+] \geq \tau$. We prefer the latter for our purposes, because it explicitly defines convergence in terms of the
387 black-box model, $\mathbf{M}(\mathbf{x})$.

388 Defining convergence in this way allows for a more intuitive interpretation of the resulting counterfactual outcomes
389 than with fixed T . Specifically, it allows us to think of counterfactuals as explaining ‘high-confidence’ predictions by
390 the model for the target class y^+ . Depending on the context and application, different choices of τ can be considered
391 as representing ‘high-confidence’ predictions.

392 **B.2 Protecting Mutability Constraints with Linear Classifiers**

393 In Section 3.2 we explain that to avoid penalizing implausibility that arises due to mutability constraints, we impose a
394 point mass prior on $p(\mathbf{x})$ for the corresponding feature. We argue in Section 3.2 that this approach induces models to
395 be less sensitive to immutable features and demonstrate this empirically in Section 4. Below we derive the analytical
396 results in Proposition 3.1.

397 *Proof.* Let d_{mtbl} and d_{immtbl} denote some mutable and immutable feature, respectively. Suppose that $\mu_{y^-, d_{\text{immtbl}}} <$
398 $\mu_{y^+, d_{\text{immtbl}}}$ and $\mu_{y^-, d_{\text{mtbl}}} > \mu_{y^+, d_{\text{mtbl}}}$, where $\mu_{k,d}$ denotes the conditional sample mean of feature d in class k . In words,
399 we assume that the immutable feature tends to take lower values for samples in the non-target class y^- than in the
400 target class y^+ . We assume the opposite to hold for the mutable feature.

401 Assuming multivariate Gaussian class densities with common diagonal covariance matrix $\Sigma_k = \Sigma$ for all $k \in \mathcal{K}$, we
402 have for the log likelihood ratio between any two classes $k, m \in \mathcal{K}$ (Hastie, Tibshirani, and Friedman 2009):

$$\log \frac{p(k|\mathbf{x})}{p(m|\mathbf{x})} = \mathbf{x}^\top \Sigma^{-1} (\mu_k - \mu_m) + \text{const} \quad (4)$$

403 By independence of x_1, \dots, x_D , the full log-likelihood ratio decomposes into:

$$\log \frac{p(k|\mathbf{x})}{p(m|\mathbf{x})} = \sum_{d=1}^D \frac{\mu_{k,d} - \mu_{m,d}}{\sigma_d^2} x_d + \text{const} \quad (5)$$

404 By the properties of our classifier (*multinomial logistic regression*), we have:

$$\log \frac{p(k|\mathbf{x})}{p(m|\mathbf{x})} = \sum_{d=1}^D (\theta_{k,d} - \theta_{m,d}) x_d + \text{const} \quad (6)$$

405 where $\theta_{k,d} = \Theta[k, d]$ denotes the coefficient on feature d for class k .

406 Based on Equation 5 and Equation 6 we can identify that $(\mu_{k,d} - \mu_{m,d}) \propto (\theta_{k,d} - \theta_{m,d})$ under the assumptions we
407 made above. Hence, we have that $(\theta_{y^-, d_{\text{immtbl}}} - \theta_{y^+, d_{\text{immtbl}}}) < 0$ and $(\theta_{y^-, d_{\text{mtbl}}} - \theta_{y^+, d_{\text{mtbl}}}) > 0$

408 Let \mathbf{x}' denote some randomly chosen individual from class y^- and let $y^+ \sim p(y)$ denote the randomly chosen target
 409 class. Then the partial derivative of the contrastive divergence penalty Equation 2 with respect to coefficient $\theta_{y^+, d}$ is
 410 equal to

$$\frac{\partial}{\partial \theta_{y^+, d}} (\text{div}(\mathbf{x}, \mathbf{x}', \mathbf{y}; \theta)) = \frac{\partial}{\partial \theta_{y^+, d}} ((-\mathbf{M}_\theta(\mathbf{x})[y^+]) - (-\mathbf{M}_\theta(\mathbf{x}')[y^+])) = x'_d - x_d \quad (7)$$

411 and equal to zero everywhere else.

412 Since $(\mu_{y^-, d_{\text{immtbl}}} < \mu_{y^+, d_{\text{immtbl}}})$ we are more likely to have $(x'_{d_{\text{immtbl}}} - x_{d_{\text{immtbl}}}) < 0$ than vice versa at initialization.
 413 Similarly, we are more likely to have $(x'_{d_{\text{mtbl}}} - x_{d_{\text{mtbl}}}) > 0$ since $(\mu_{y^-, d_{\text{mtbl}}} > \mu_{y^+, d_{\text{mtbl}}})$.

414 This implies that if we do not protect feature d_{immtbl} , the contrastive divergence penalty will decrease $\theta_{y^-, d_{\text{immtbl}}}$ thereby
 415 exacerbating the existing effect $(\theta_{y^-, d_{\text{immtbl}}} - \theta_{y^+, d_{\text{immtbl}}}) < 0$. In words, not protecting the immutable feature would have
 416 the undesirable effect of making the classifier more sensitive to this feature, in that it would be more likely to predict
 417 class y^- as opposed to y^+ for lower values of d_{immtbl} .

418 By the same rationale, the contrastive divergence penalty can generally be expected to increase $\theta_{y^-, d_{\text{mtbl}}}$ exacerbating
 419 $(\theta_{y^-, d_{\text{mtbl}}} - \theta_{y^+, d_{\text{mtbl}}}) > 0$. In words, this has the effect of making the classifier more sensitive to the mutable feature, in
 420 that it would be more likely to predict class y^- as opposed to y^+ for higher values of d_{mtbl} .

421 Thus, our proposed approach of protecting feature d_{immtbl} has the net affect of decreasing the classifier's sensitivity
 422 to the immutable feature relative to the mutable feature (i.e. no change in sensitivity for d_{immtbl} relative to increased
 423 sensitivity for d_{mtbl}). \square

⚠ Warning

@Cynthia, @Arie, I have tentatively phrased the above in terms of a theorem and proof. This is something I've so far shied away from because I feel a bit out of my depth when it comes to mathematical proofs. The above makes intuitive sense to me, but I don't know for sure if it's correct.

424

425 B.3 Domain Constraints

426 We apply domain constraints on counterfactuals during training and evaluation. There are at least two good reasons for
 427 doing so. Firstly, within the context of explainability and algorithmic recourse, real-world attributes are often domain
 428 constrained: the *age* feature, for example, is lower bounded by zero and upper bounded by the maximum human
 429 lifespan. Secondly, domain constraints help mitigate training instabilities commonly associated with energy-based
 430 modelling (Grathwohl et al. 2020; Altmeyer et al. 2024).

431 For our image datasets, features are pixel values and hence the domain is constrained by the lower and upper bound
 432 of values that pixels can take depending on how they are scaled (in our case $[-1, 1]$). For all other features d in our
 433 synthetic and tabular datasets, we automatically infer domain constraints $[x_d^{\text{LB}}, x_d^{\text{UB}}]$ as follows,

$$\begin{aligned} x_d^{\text{LB}} &= \arg \min_{x_d} \{\mu_d - n_{\sigma_d} \sigma_d, \arg \min_{x_d} x_d\} \\ x_d^{\text{UB}} &= \arg \max_{x_d} \{\mu_d + n_{\sigma_d} \sigma_d, \arg \max_{x_d} x_d\} \end{aligned} \quad (8)$$

434 where μ_d and σ_d denote the sample mean and standard deviation of feature d . We set $n_{\sigma_d} = 3$ across the board but
 435 higher values and hence wider bounds may be appropriate depending on the application.

436 B.4 Training Details

437 In this section, we describe the training procedure in detail. While the details laid out here are not crucial for under-
 438 standing our proposed approach, they are of importance to anyone looking to implement counterfactual training.

439 **C Detailed Results**

440 **⚠ Warning**

441 @Cynthia, @Arie, I'm including some preliminary results here but will rerun experiments in the coming days.
 442 You'll notice some odd outlier results (huge values) for the initial grid search (Section C.2). My belief is that
 443 this is once again due to *ECCo* overshooting for some hyperparameter choices, that we have discussed before.
 444 A simple solution to this story is to actually impose domain constraints. Let's see what the final results show us
 445 (I also still plan to make slight changes to the implementation), but in any case I think it might even be worth
 446 to report results for the initial grid search in the final appendix (they do include good results for CT for certain
 447 hyperparameter ranges and highlight limitations inherited from energy-based modelling).

448 **C.1 Qualitative Findings for Image Data**

449 **ℹ Note**

450 @Cynthia, @Arie, Figure A1 shows much more plausible (faithful) counterfactuals for a model with CT than
 451 the model with conventional training (Figure A2). In fact, this is not even using *ECCo+* and still showing better
 452 results than the best results we achieved in our AAAI paper for JEM ensembles.

453 **C.2 Initial Grid Search**

454 For the initial round of experiments we employed a different training objective and procedure that led to promising
 455 results for some hyperparameter choices, but suffered from training instabilities.

456 **C.2.1 Generator Parameters**

457 The hyperparameter grids for the first investigation of the effect of generator parameters are shown in Parameters C.1
 458 and Parameters C.2.

459 **Parameters C.1 (Training Phase).**

- 460 • Generator Parameters:
 - 461 – Decisiono. Threshold: 0.75, 0.9, 0.95
 - 462 – λ_{cost} : 0.0, 0.1
 - 463 – λ_{div} : 0.1, 0.5, 1.0, 10.0
 - 464 – Maximum Iterations: 5, 25, 50
 - 465 – Optimizerimizer: sgd
- 466 • Generator: ecco, generic, revise
- 467 • Training Parameters:
 - 468 – Objective: full, vanilla

469 **Parameters C.2 (Evaluation Phase).**

- 470 • Counterfactual Parameters:
 - 471 – Convergence: max_iter
 - 472 – Maximum Iterations: 100
 - 473 – No. Individuals: 100
 - 474 – No. Runs: 5
- 475 • Generator Parameters:
 - 476 – λ_{cost} : 0.0
 - 477 – λ_{div} : 0.1, 0.5, 1.0, 5.0, 10.0, 20.0
 - 478 – Learning Rate: 1.0
 - 479 – Maximum Iterations: 50
 - 480 – Optimizerimizer: sgd

481 **C.2.1.1 Linearly Separable**

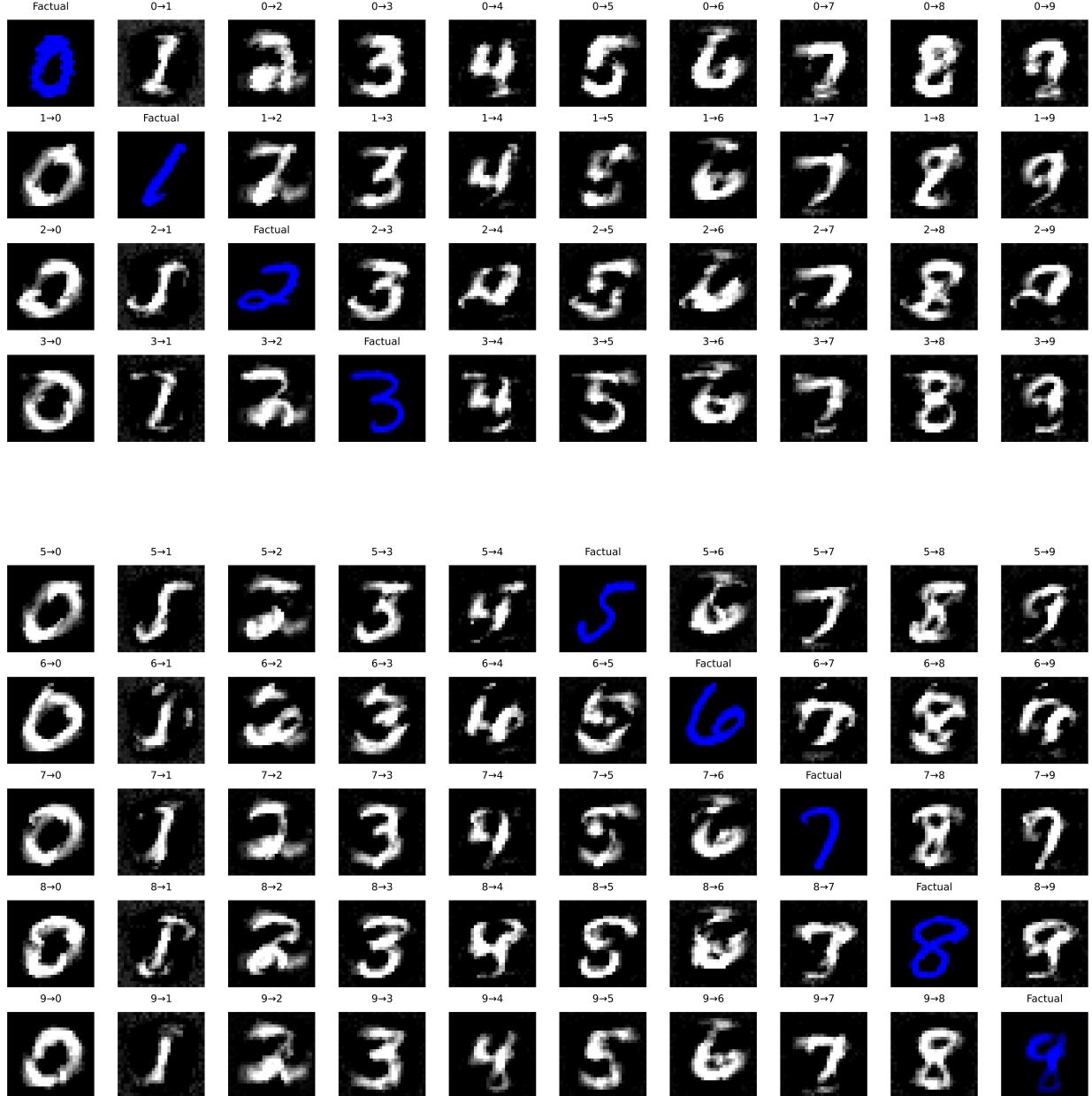


Figure A1: Counterfactual images for *MLP* with counterfactual training. The underlying generator, *ECCo*, aims to generate counterfactuals that are faithful to the model (Altmeyer et al. 2024).

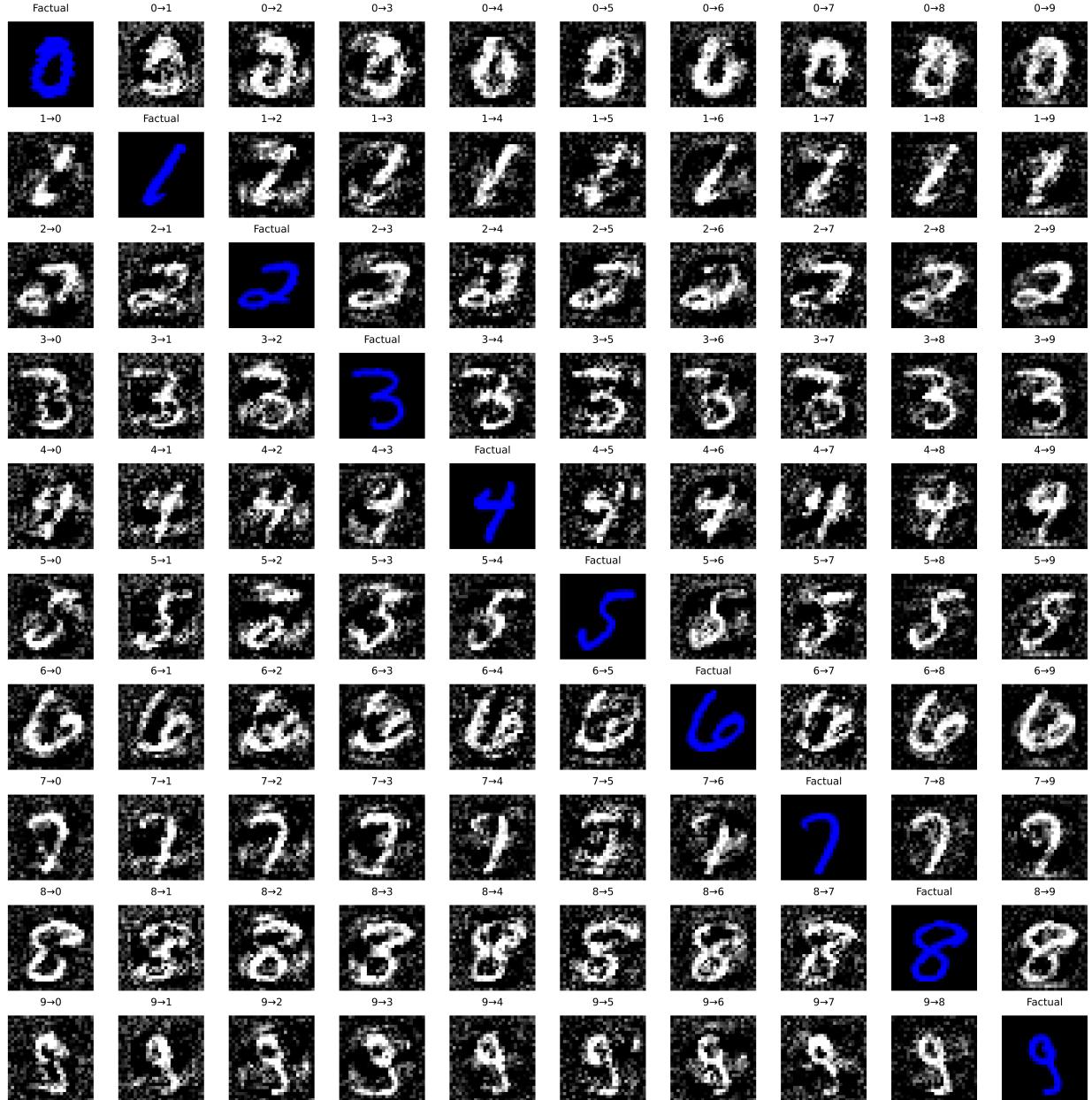


Figure A2: Counterfactual images for *MLP* with conventional training. The underlying generator, *ECCo*, aims to generate counterfactuals that are faithful to the model (Altmeyer et al. 2024).

- 473 • **Energy Penalty** (Table A1): *ECCo* generally does yield better results than *Vanilla* for higher choices of the
 474 energy penalty (10,15) during training. *Generic* performs poorly across the board. *Omni* seems to have an
 475 anchoring effect, in that it never performs terribly but also never as good as the best *ECCo* results. *REVISE*
 476 performs poorly across the board.
- 477 • **Cost** (Table A2): Results for all generators (except *Omni*) are quite bad, which can likely be attributed to
 478 extremely bad results for some choices of the **Energy Penalty** (results here are averaged). For *ECCo* and
 479 *Generic*, higher cost values generally lead to worse results.
- 480 • **Maximum Iterations**: No clear patterns recognizable, so it seems that smaller choices are ok.
- 481 • **Validity**: *ECCo* almost always valid except for very low values during training and high values at evaluation
 482 time. *Generic* often has poor validity.
- 483 • **Accuracy**: Seems largely unaffected.

Table A1: Results for Linearly Separable data by energy penalty.

Objective	$\lambda_{\text{div}}(\text{train})$	Generator	Value	Std
full	0.01	<i>ECCo</i>	$-9.91 \cdot 10^{11}$	$2.25 \cdot 10^{12}$
full	0.01	<i>Generic</i>	$-5.71 \cdot 10^{17}$	$1.3 \cdot 10^{18}$
full	0.01	REVISE	-15.6	13.2
vanilla	0.01	<i>ECCo</i>	-4.28	3.52
vanilla	0.01	<i>Generic</i>	-4.45	3.47
vanilla	0.01	<i>REVISE</i>	-4.91	4.24
full	0.05	<i>ECCo</i>	$-5.63 \cdot 10^5$	$1.28 \cdot 10^6$
full	0.05	<i>Generic</i>	$-8.35 \cdot 10^{17}$	$1.9 \cdot 10^{18}$
full	0.05	<i>REVISE</i>	-15	12.6
vanilla	0.05	<i>ECCo</i>	-4.4	3.66
vanilla	0.05	Generic	-4.38	3.48
vanilla	0.05	<i>REVISE</i>	-4.94	4.22
full	0.1	<i>ECCo</i>	$-6.74 \cdot 10^5$	$1.53 \cdot 10^6$
full	0.1	<i>Generic</i>	$-1.71 \cdot 10^{11}$	$3.9 \cdot 10^{11}$
full	0.1	<i>REVISE</i>	-15.6	13.2
vanilla	0.1	<i>ECCo</i>	-4.28	3.52
vanilla	0.1	<i>Generic</i>	-4.45	3.48
vanilla	0.1	<i>REVISE</i>	-4.91	4.25
full	0.5	ECCo	-11.8	9.83
full	0.5	<i>Generic</i>	$-1.06 \cdot 10^{18}$	$2.42 \cdot 10^{18}$
full	0.5	<i>REVISE</i>	-15	12.6
vanilla	0.5	<i>ECCo</i>	-4.4	3.65
vanilla	0.5	<i>Generic</i>	-4.38	3.48
vanilla	0.5	<i>REVISE</i>	-4.95	4.22
full	1	<i>ECCo</i>	-11.5	11.1
full	1	<i>Generic</i>	$-1.71 \cdot 10^{11}$	$3.88 \cdot 10^{11}$
full	1	REVISE	-15.7	13.3
vanilla	1	<i>ECCo</i>	-4.28	3.51
vanilla	1	<i>Generic</i>	-4.44	3.47
vanilla	1	<i>REVISE</i>	-4.91	4.25
full	5	<i>ECCo</i>	-3.99	3.12
full	5	<i>Generic</i>	$-4.88 \cdot 10^{17}$	$1.11 \cdot 10^{18}$
full	5	<i>REVISE</i>	-14.6	12.1
vanilla	5	<i>ECCo</i>	-4.4	3.65
vanilla	5	Generic	-4.38	3.48
vanilla	5	<i>REVISE</i>	-4.95	4.22
full	10	<i>ECCo</i>	-2.31	0.735
full	10	<i>Generic</i>	$-1.7 \cdot 10^{11}$	$3.86 \cdot 10^{11}$
full	10	<i>REVISE</i>	-15.5	13
vanilla	10	<i>ECCo</i>	-4.28	3.51
vanilla	10	<i>Generic</i>	-4.44	3.47

Continuing table below.

Objective	$\lambda_{\text{div}}(\text{train})$	Generator	Value	Std
vanilla	10	<i>REVISE</i>	-4.91	4.24
full	15	ECCo	-2.01	0.488
full	15	<i>Generic</i>	$-4.91 \cdot 10^{17}$	$1.12 \cdot 10^{18}$
full	15	<i>REVISE</i>	-14.4	11.7
vanilla	15	<i>ECCo</i>	-4.4	3.65
vanilla	15	<i>Generic</i>	-4.38	3.48
vanilla	15	<i>REVISE</i>	-4.95	4.23

Table A2: Results for Linearly Separable data by cost penalty.

Objective	$\lambda_{\text{cost}}(\text{train})$	Generator	Value	Std
full	0	<i>ECCo</i>	$-5.32 \cdot 10^3$	$1.21 \cdot 10^4$
full	0	<i>Generic</i>	$-1.03 \cdot 10^{18}$	$2.34 \cdot 10^{18}$
full	0	REVISE	-15.4	12.9
vanilla	0	<i>ECCo</i>	-4.34	3.58
vanilla	0	<i>Generic</i>	-4.41	3.48
vanilla	0	<i>REVISE</i>	-4.93	4.23
full	0.001	<i>ECCo</i>	-362	811
full	0.001	<i>Generic</i>	$-2.65 \cdot 10^{17}$	$6.04 \cdot 10^{17}$
full	0.001	<i>REVISE</i>	-15.5	13
vanilla	0.001	<i>ECCo</i>	-4.34	3.58
vanilla	0.001	Generic	-4.41	3.48
vanilla	0.001	<i>REVISE</i>	-4.93	4.23
full	0.1	<i>ECCo</i>	$-3.72 \cdot 10^{11}$	$8.46 \cdot 10^{11}$
full	0.1	<i>Generic</i>	$-4.48 \cdot 10^{14}$	$1.02 \cdot 10^{15}$
full	0.1	<i>REVISE</i>	-14.6	12.2
vanilla	0.1	<i>ECCo</i>	-4.34	3.58
vanilla	0.1	<i>Generic</i>	-4.41	3.48
vanilla	0.1	<i>REVISE</i>	-4.93	4.24

484 C.2.1.2 Moons

- 485 • **Energy Penalty** (Table A3): *ECCo* consistently yields better results than *Vanilla*, except for very low choices
486 of the energy penalty during training for which it performs abysmal. *Generic* performs quite badly across
487 the board for high enough choices of the energy penalty at evaluation time. *Omni* has small positive effect.
488 *REVISE* performs poorly across the board.
- 489 • **Cost (distance penalty)**: *Generic* generally does better for higher values, while *ECCo* does better for lower
490 values.
- 491 • **Maximum Iterations**: No clear patterns recognizable, so it seems that smaller choices are ok.
- 492 • **Validity**: *ECCo* generally achieves full validity except for very low choices the energy penalty during training
493 and high choices at evaluation time. *Generic* performs poorly for high choices of the energy penalty during
494 evaluation.
- 495 • **Accuracy**: Largely unaffected although *ECCo* suffers a bit for very low choices the energy penalty during
496 training. *REVISE* suffers a lot in general (around 10 percentage points).

Table A3: Results for Moons data by energy penalty.

Objective	$\lambda_{\text{div}}(\text{train})$	Generator	Value	Std
full	0.01	<i>ECCo</i>	$-2.8 \cdot 10^{22}$	$6.39 \cdot 10^{22}$
full	0.01	<i>Generic</i>	$-4.89 \cdot 10^{30}$	$1.11 \cdot 10^{31}$
full	0.01	REVISE	-572	$1.25 \cdot 10^3$
vanilla	0.01	<i>ECCo</i>	-15.5	17.3

Continuing table below.

Objective	$\lambda_{\text{div}}(\text{train})$	Generator	Value	Std
vanilla	0.01	<i>Generic</i>	-10.9	11.9
vanilla	0.01	<i>REVISE</i>	-11.2	13
full	0.05	<i>ECCo</i>	$-1.55 \cdot 10^{16}$	$3.52 \cdot 10^{16}$
full	0.05	<i>Generic</i>	$-2.22 \cdot 10^{20}$	$5 \cdot 10^{20}$
full	0.05	<i>REVISE</i>	$-1.04 \cdot 10^3$	$2.3 \cdot 10^3$
vanilla	0.05	<i>ECCo</i>	-15.5	17.2
vanilla	0.05	Generic	-11.7	12.8
vanilla	0.05	<i>REVISE</i>	-11.3	13.1
full	0.1	<i>ECCo</i>	$-3.41 \cdot 10^3$	$7.73 \cdot 10^3$
full	0.1	<i>Generic</i>	$-5.22 \cdot 10^{30}$	$1.19 \cdot 10^{31}$
full	0.1	<i>REVISE</i>	-288	594
vanilla	0.1	<i>ECCo</i>	-15.5	17.2
vanilla	0.1	<i>Generic</i>	-10.9	11.9
vanilla	0.1	<i>REVISE</i>	-11.3	13.1
full	0.5	ECCo	-7.09	7.51
full	0.5	<i>Generic</i>	$-1.11 \cdot 10^{31}$	$2.53 \cdot 10^{31}$
full	0.5	<i>REVISE</i>	$-1.19 \cdot 10^3$	$2.64 \cdot 10^3$
vanilla	0.5	<i>ECCo</i>	-15.5	17.2
vanilla	0.5	<i>Generic</i>	-11.7	12.8
vanilla	0.5	<i>REVISE</i>	-11.3	13.1
full	1	<i>ECCo</i>	-6.06	6.33
full	1	<i>Generic</i>	$-1.58 \cdot 10^{33}$	$3.59 \cdot 10^{33}$
full	1	REVISE	$-1.16 \cdot 10^3$	$2.59 \cdot 10^3$
vanilla	1	<i>ECCo</i>	-15.5	17.3
vanilla	1	<i>Generic</i>	-10.9	11.9
vanilla	1	<i>REVISE</i>	-11.3	13.1
full	5	<i>ECCo</i>	-2.57	2.07
full	5	<i>Generic</i>	$-1.17 \cdot 10^{28}$	$2.66 \cdot 10^{28}$
full	5	<i>REVISE</i>	-530	$1.16 \cdot 10^3$
vanilla	5	<i>ECCo</i>	-15.5	17.2
vanilla	5	Generic	-11.7	12.7
vanilla	5	<i>REVISE</i>	-11.3	13.1
full	10	<i>ECCo</i>	-1.76	0.974
full	10	<i>Generic</i>	$-1.54 \cdot 10^{33}$	$3.51 \cdot 10^{33}$
full	10	<i>REVISE</i>	$-1.52 \cdot 10^3$	$3.4 \cdot 10^3$
vanilla	10	<i>ECCo</i>	-15.5	17.3
vanilla	10	Generic	-10.9	11.9
vanilla	10	<i>REVISE</i>	-11.3	13.1
full	15	<i>ECCo</i>	-1.37	0.365
full	15	<i>Generic</i>	$-5.32 \cdot 10^{28}$	$1.21 \cdot 10^{29}$
full	15	<i>REVISE</i>	-473	$1.03 \cdot 10^3$
vanilla	15	<i>ECCo</i>	-15.5	17.2
vanilla	15	<i>Generic</i>	-11.7	12.8
vanilla	15	<i>REVISE</i>	-11.3	13.1

497 C.2.1.3 Circles

- 498 • **Energy Penalty** (Table A4): *ECCo* consistently yields better results than *Vanilla*, though primarily for low to
 499 medium choices of the energy penalty ($<=5$) during training. The same goes for *Generic*, which sometimes
 500 outperforms *ECCo* (for small energy penalty at evaluation time). *Omni* does alright for lower energy penalty
 501 at evaluation time, but loses out for higher choices. *REVISE* performs poorly across the board (except very
 502 low choices at evaluation time).
- 503 • **Cost (distance penalty)**: *ECCo* and *Generic* generally achieve the best results when no cost penalty is used
 504 during training. Both *Omni* and *REVISE* are largely unaffected.
- 505 • **Maximum Iterations**: *ECCo* consistently yields better results for higher numbers of iterations. *Generic*
 506 generally does best for a medium number (50). *Omni* is sometimes invalid (???).

- 507 • **Validity:** *ECCo* tends to outperform its *Vanilla* counterpart, though primarily for low to medium choices of
 508 the energy penalty ($<=5$) during training and evaluation. *Vanilla* typically worse across the board.
 509 • **Accuracy:** Mostly unaffected, but *REVISE* again consistently some deterioration and *ECCo* deteriorates for
 510 high choices of energy penalty during training, reflecting other outcomes above.

Table A4: Results for Circles data by energy penalty.

Objective	$\lambda_{\text{div}}(\text{train})$	Generator	Value	Std
full	0.01	ECCo	-1.26	0.423
full	0.01	<i>Generic</i>	-1.49	0.71
full	0.01	<i>REVISE</i>	$-2.71 \cdot 10^{26}$	$6.37 \cdot 10^{26}$
vanilla	0.01	<i>ECCo</i>	-9.33	7.34
vanilla	0.01	<i>Generic</i>	-8.89	6.88
vanilla	0.01	<i>REVISE</i>	-8.65	6.8
full	0.05	<i>ECCo</i>	-1.29	0.397
full	0.05	<i>Generic</i>	-1.21	0.356
full	0.05	<i>REVISE</i>	$-5.91 \cdot 10^{27}$	$1.36 \cdot 10^{28}$
vanilla	0.05	ECCo	-9.35	7.32
vanilla	0.05	<i>Generic</i>	-8.85	6.87
vanilla	0.05	<i>REVISE</i>	-8.52	6.76
full	0.1	<i>ECCo</i>	-1.2	0.383
full	0.1	<i>Generic</i>	-1.5	0.735
full	0.1	<i>REVISE</i>	$-3.06 \cdot 10^{26}$	$7.7 \cdot 10^{26}$
vanilla	0.1	<i>ECCo</i>	-9.33	7.32
vanilla	0.1	Generic	-8.88	6.86
vanilla	0.1	<i>REVISE</i>	-8.68	6.81
full	0.5	<i>ECCo</i>	-1.12	0.217
full	0.5	<i>Generic</i>	-1.21	0.352
full	0.5	<i>REVISE</i>	$-5.97 \cdot 10^{27}$	$1.37 \cdot 10^{28}$
vanilla	0.5	<i>ECCo</i>	-9.35	7.3
vanilla	0.5	<i>Generic</i>	-8.89	6.92
vanilla	0.5	<i>REVISE</i>	-8.53	6.75
full	1	ECCo	-1.1	0.163
full	1	<i>Generic</i>	-1.49	0.726
full	1	<i>REVISE</i>	$-3.09 \cdot 10^{26}$	$7.22 \cdot 10^{26}$
vanilla	1	<i>ECCo</i>	-9.34	7.36
vanilla	1	<i>Generic</i>	-8.86	6.85
vanilla	1	<i>REVISE</i>	-8.69	6.85
full	5	<i>ECCo</i>	-1.75	0.154
full	5	<i>Generic</i>	-1.21	0.363
full	5	REVISE	$-1.1 \cdot 10^{28}$	$2.5 \cdot 10^{28}$
vanilla	5	<i>ECCo</i>	-9.36	7.32
vanilla	5	<i>Generic</i>	-8.88	6.91
vanilla	5	<i>REVISE</i>	-8.52	6.73
full	10	<i>ECCo</i>	$-1.02 \cdot 10^6$	$2.32 \cdot 10^6$
full	10	<i>Generic</i>	-1.49	0.702
full	10	<i>REVISE</i>	$-3.74 \cdot 10^{26}$	$9.09 \cdot 10^{26}$
vanilla	10	<i>ECCo</i>	-9.31	7.33
vanilla	10	<i>Generic</i>	-8.87	6.86
vanilla	10	REVISE	-8.69	6.83
full	15	<i>ECCo</i>	$-3.31 \cdot 10^{13}$	$7.54 \cdot 10^{13}$
full	15	<i>Generic</i>	-1.22	0.37
full	15	<i>REVISE</i>	$-9.01 \cdot 10^{27}$	$2.06 \cdot 10^{28}$
vanilla	15	<i>ECCo</i>	-9.38	7.34
vanilla	15	<i>Generic</i>	-8.86	6.87
vanilla	15	<i>REVISE</i>	-8.51	6.73

511 **D Computation Details**

512 **D.1 Hardware**

513 We performed our experiments on a high-performance cluster. Details about the cluster will be disclosed upon publi-
 514 cation to avoid revealing information that might interfere with the double-blind review process. Since our experiments
 515 have largely involved highly parallel tasks and small models, we have made extensive use of multi-processing.

516 **D.2 Software**

517 All computations were performed in the Julia Programming Language ([Bezanson et al. 2017](#)). We have developed
 518 a package for counterfactual training that leverages and extends the functionality provided by several existing pack-
 519 ages, most notably [CounterfactualExplanations.jl](#) ([Altmeyer, Deursen, et al. 2023](#)) and the [Flux.jl](#) library for deep
 520 learning ([Michael Innes et al. 2018; Mike Innes 2018](#)). For data-wrangling and presentation-ready tables we relied on
 521 [DataFrames.jl](#) ([Bouchet-Valat and Kamiski 2023](#)) and [PrettyTables.jl](#) ([Chagas et al. 2024](#)), respectively. For plots and
 522 visualizations we used both [Plots.jl](#) ([Christ et al. 2023](#)) and [Makie.jl](#) ([Danisch and Krumbiegel 2021](#)), in particular
 523 [AlgebraOfGraphics.jl](#).