CF-OPT: Counterfactual Explanations for Structured Prediction

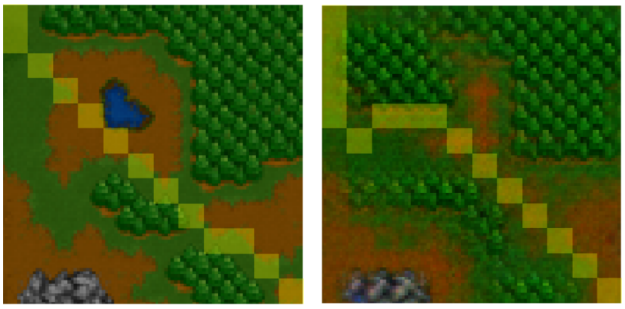
Germain Vivier-Ardisson [1 2] **Alexandre Forel** [1] **Axel Parmentier** [2] **Thibaut Vidal** [1]

Abstract

Optimization layers in deep neural networks have enjoyed a growing popularity in structured learning, improving the state of the art on a variety of applications. Yet, these pipelines lack interpretability since they are made of two opaque layers: a highly non-linear prediction model, such as a deep neural network, and an optimization layer, which is typically a complex black-box solver. Our goal is to improve the transparency of such methods by providing counterfactual explanations. We build upon variational autoencoders a principled way of obtaining counterfactuals: working in the latent space leads to a natural notion of plausibility of explanations. We finally introduce a variant of the classic loss for VAE training that improves their performance in our specific structured context. These provide the foundations of `CF-OPT`, a first-order optimization algorithm that can find counterfactual explanations for a broad class of structured learning architectures. Our numerical results show that both close and plausible explanations can be obtained for problems from the recent literature.

1. Introduction

Recent studies have shown a surge of interest in developing structured learning pipelines that combine machine learning and optimization (Amos & Kolter, 2017; Donti et al., 2017). From the perspective of structured learning, optimization layers translate a machine-learning prediction into an output that maximizes a given objective while respecting structural constraints. From an optimization perspective, machine-learning models can identify patterns in data to optimize an objective function subject to uncertain parameters. While these pipelines have shown great results in many applications (Mandi et al., 2023; Sadana et al., 2023), it is hard to justify their outputs since both their prediction and optimization components are complex and opaque. This lack of interpretability is hardly tackled, even though it is likely to hinder the adoption of this type of solution. We aim to explain the decision-making process of structured learning pipelines by providing counterfactual explanations. Counterfactual explanations show the minimum changes in a feature vector needed to lead the pipeline to output a different decision. By contrasting the initial factual instance with its counterfactual explanation, they allow users to understand the impact of features on the decision. Indeed, identifying which features make a specific solution optimal and establishing a link between changes in the features and changes in the output are essential conditions for the reliable use of a model.



(a) Initial map (b) Counterfactual map

Figure 1. (a) Initial and (b) counterfactual maps with their respective shortest path (initial and alternative solutions) shown in yellow. The explanation is given by `CF-OPT`. The corresponding pipeline and experiment is detailed in Section 5.1 and will serve as a guiding example in this work.

In this paper, we present `CF-OPT`, a method to obtain counterfactual explanations for any deep learning model concluded by a black-box linear optimization layer, called thereafter structured learning pipeline. It is based on differentiating a relaxed version of the explanation problem and applying a first-order optimization method to obtain a counterfactual. An example explanation provided by `CF-OPT` is shown in Figure 1 for the shortest paths on Warcraft maps structured learning pipeline, taken from Vlastelica et al. (2019).

[1]CIRRELT & SCALE-AI Chair in Data-Driven Supply Chains, Department of Mathematical and Industrial Engineering, Polytechnique Montreal, Montreal, Canada [2]CERMICS, École des Ponts, Marne-la-Vallée, France. Correspondence to: Germain Vivier-Ardisson <germain.vivier-ardisson@enpc.fr>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

We make the following contributions:

(i) We present an efficient algorithm to provide counterfactual explanations to structured learning pipelines that combine a deep model and an optimization model. It is based on an augmented Lagrangian relaxation of the explanation problem and a first-order optimization algorithm.

(ii) We propose a new type of explanation that does not require a target decision and can be used for local sensitivity analyses.

(iii) We highlight the sensitivity of structured learning pipelines to subtle perturbations: for high-dimensional inputs, naive counterfactual explanations are equivalent to adversarial attacks. As a consequence, we define a notion of plausibility region and develop a method to obtain plausible explanations even in high-dimensional settings thanks to a Variational Auto-Encoder (VAE).

(iv) We take advantage of the prior imposed on the latent space of the VAE in order to add a plausibility regularization term to the explanation objective. Furthermore, we modify the training objective of the VAE by introducing a cost-aware loss, designed to account for the downstream optimization task.

(v) We demonstrate the value of our method and analyze its principal components thanks to numerical experiments. We show that we can compute explanations efficiently, even for large problems.

2. Problem Statement

In this paper, we consider structured learning models of the form

$$x \longmapsto y^*(x) \in \operatorname*{argmin}_{y \in \mathcal{Y}} \varphi(x)^\top y, \tag{1}$$

where $x \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$ is a vector of features, equivalently referred to as *context* or *covariate*. The prediction model $\varphi : \mathcal{X} \to \Theta$ transforms x into an intermediate input $\varphi(x) = \theta \in \Theta \subseteq \mathbb{R}^{n_\theta}$. This intermediate input parameterizes the linear optimization layer, which returns the decision $y^* \in \operatorname{argmin}_{y \in \mathcal{Y}} \theta^\top y$, where $\mathcal{Y} \subseteq \mathbb{R}^{n_y}$ is the feasible set. A schematic representation is given in Figure 2.

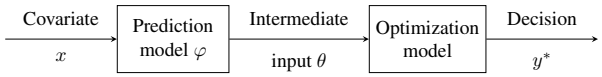


Figure 2. Structured learning pipeline.

We focus on optimization layers of feasible set $\mathcal{Y}$ defined by linear with possible integrality constraints, and make no assumption on the solver used to compute y^*.

This setting encompasses a wide array of methods proposed in recent years, used to make structured predictions (Wilder et al., 2019; Vlastelica et al., 2019; Ferber et al., 2020; Berthet et al., 2020; Niepert et al., 2021; Sahoo et al., 2023; Stewart et al., 2023), to solve contextual stochastic problems whose objective is linear in the uncertain parameters (Elmachtoub & Grigas, 2022; Jeong et al., 2022; McKenzie et al., 2023), or to approximate hard optimization problems by learning linear surrogate models (Dalle et al., 2022; Ferber et al., 2023; Gupta & Zhang, 2024). While these works differ in the way the models are trained, the structure of the resulting pipeline, in particular the optimization layer, is identical to the one in Figure 2. This work thus allows to improve the interpretability of a large array of recent methods. The only assumption we make on the prediction model is that it is differentiable w.r.t to its input, so that φ can typically be any state-of-the-art deep learning model. Hence, our setting extends the one of Forel et al. (2023), who provide explanations for a restricted type of pipeline based on random forests and k-nearest neighbors.

2.1. Counterfactual Explanations of Decisions

Counterfactual explanations of decisions are based on contrasting a covariate with an alternative one. Suppose that, at decision time, we observe covariate x_0 and evaluate the trained pipeline to obtain decision y_0. Let y^{alt} be an alternative decision representing, for instance, the solution usually chosen by an expert in context x_0. We explain the decision y_0 by comparing its covariate x_0 to an alternative covariate x^{alt} that we find, i.e., the counterfactual explanation. Let $\theta^{\text{alt}} = \varphi(x^{\text{alt}})$ be the output of the prediction model for this alternative covariate. Two types of explanation have been introduced by Forel et al. (2023).

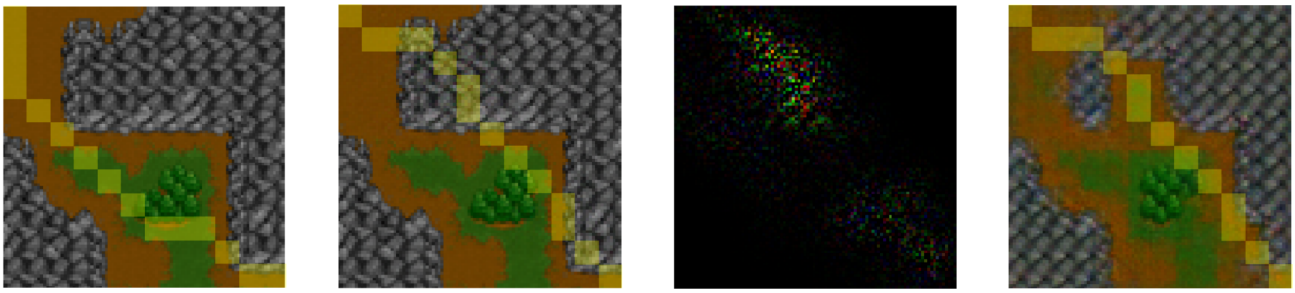
Definition 2.1. A covariate x^{alt} is a ***relative explanation*** if the decision y^{alt} is at least as good as y_0 in the context x^{alt}, that is: $\theta^{\text{alt}\top} y^{\text{alt}} \leq \theta^{\text{alt}\top} y_0$.

Definition 2.2. A covariate x^{alt} is an ***absolute explanation*** if the decision y^{alt} is optimal in the context x^{alt}, that is: $y^{\text{alt}} \in \operatorname{argmin}_{y \in \mathcal{Y}} \theta^{\text{alt}\top} y$.

Definitions 2.1 and 2.2 require that the practitioner provides an alternative solution y^{alt}. When such a solution is not available, the decision-maker may be interested in analyzing the local sensitivity of the decision y_0. To this end, we propose a third, novel type of explanation, aiming at finding a context x^{alt} in which y_0 is not optimal anymore. To control the amount of suboptimality of the initial solution y_0 in the new context x^{alt}, we add a parameter ε.

Definition 2.3. A covariate x^{alt} is an ***ε-explanation*** if $\min_{y \in \mathcal{Y}} \theta^{\text{alt}\top} y + \varepsilon \,|\min_{y \in \mathcal{Y}} \theta^{\text{alt}\top} y| \leq \theta^{\text{alt}\top} y_0$, that is, y_0 has a relative optimality gap (or relative regret) of at least ε in the context x^{alt}.

Remark 2.4. For the sake of clarity, we will consider below that the costs of solutions are necessarily positive (which is true, e.g., in our guiding example on Warcraft maps). The condition for x^{alt} being an ε-explanation then reduces to $(1+\varepsilon) \min_{y \in \mathcal{Y}} \theta^{\text{alt}\top} y \leq \theta^{\text{alt}\top} y_0$.



(a) Initial map and its associated shortest path

(b) Adversarial explanation obtained with naive algorithm

(c) Magnified difference (x10) between (a) and (b)

(d) Plausible explanation obtained with CF-OPT

Figure 3. Naive counterfactual search in raw feature space leads to adversarial examples. CF-OPT recovers a plausible explanation, using a VAE trained in a cost-aware fashion, and a latent hypersphere plausibility regularized objective ($\alpha = 2, \beta = 10$).

In order to facilitate the understanding of these definitions, we present several examples of relative, absolute and ε-explanations obtained with CF-OPT for the Warcraft Maps structured learning pipeline in Appendix C.

2.2. The Explanation Problem

Several desirable properties have been identified for counterfactual explanations (Wachter et al., 2017; Verma et al., 2020). Arguably, the two most important ones are proximity and plausibility. The first property ensures that the counterfactual is close to the initial context, and the latter ensures that it is close to the data manifold.

We introduce the function h to measure the satisfaction of the explanation criterion, that is:

$$h(x) = \begin{cases} \varphi(x)^\top \left(y^{\text{alt}} - y_0\right) \text{ for relative explanations,} \\ \varphi(x)^\top \left(y^{\text{alt}} - y^*(x)\right) \text{ for absolute explanations,} \\ \varphi(x)^\top \left((1+\varepsilon)\, y^*(x) - y_0\right) \text{ for } \varepsilon\text{-explanations.} \end{cases}$$

A context x is a valid explanation if $h(x) \leq 0$. Without loss of generality, we consider thereafter the explanation constraint to be an equality, that is $h(x) = 0$, which is always possible by introducing a slack variable.

Constrained optimization. Let ℓ be a differentiable function used to measure proximity in the contextual feature space $\mathcal{X}$, e.g., the squared Euclidean distance. A close and plausible counterfactual explanation x^{alt} is a solution to the constrained optimization problem given by:

$$\min_{x \in \mathcal{X}} \quad \ell(x_0, x) \tag{2a}$$

$$\text{s.t.} \quad h(x) = 0, \tag{2b}$$

$$x \in \mathcal{D} \tag{2c}$$

where $\mathcal{D}$ is a *region of plausibility*.

Region of plausibility The counterfactual explanation literature has not converged on a definition of plausibility. We outline the following desirable properties for such a region. (i) A region $\mathcal{D}$ that allows searching for close explanations by having small expected distance to initial contexts, i.e., $\mathbb{E}_{x_0 \sim \mathbb{P}_{\text{data}}}\left[\min_{x \in \mathcal{D}} \ell(x_0, x)\right]$ is small; (ii) A low volume $\text{Leb}(\mathcal{D})$ w.r.t the Lebesgue measure in order to avoid the presence of low probability areas within $\mathcal{D}$; (iii) Respecting the symmetries of the problem: e.g, if $\mathbb{P}_{\text{data}}$ is isotropic, $\mathcal{D}$ should also be. The constraint $x \in \mathcal{D}$ therefore models the plausibility of the explanation, and is implicit in the sense that the distribution $\mathbb{P}_{\text{data}}$ is unknown. However, it is critical to ensure that the explanation is not an adversarial example as illustrated in Figure 3. We provide a method to build such a region in the next section.

The link between adversarial examples and counterfactual explanations is well documented in the predictive setting, where one aims to explain classifiers (Browne & Swift, 2020; Pawelczyk et al., 2022; Freiesleben, 2022). Both consist in finding the closest input that flips a model's prediction and share the same optimization formulation. Our setting is more complex since the model to be explained is a structured learning pipeline and not a simple classifier. Still, there are clear connections between explanations of decisions and adversarial examples. Absolute explanations present a constrained optimization formulation close to that of white-box targeted adversarial attacks, that aim at finding the closest input that makes the output probability of a given class the highest one (Ma et al., 2021). Likewise, ε-explanations are similar to untargeted adversarial attacks.

3. Modeling Plausibility with a Variational Autoencoder

In CF-OPT, we use a VAE to get a meaningful and tractable formulation of the plausibility constraint $x \in \mathcal{D}$. More precisely, (i) we reformulate the plausibility constraint in the latent space, and (ii) we relax it into a soft constraint via a regularized objective. This strategy is made even more efficient by (iii) tailoring the VAE learning algorithm to obtain a model better adapted to modeling plausibility in our structured setting.

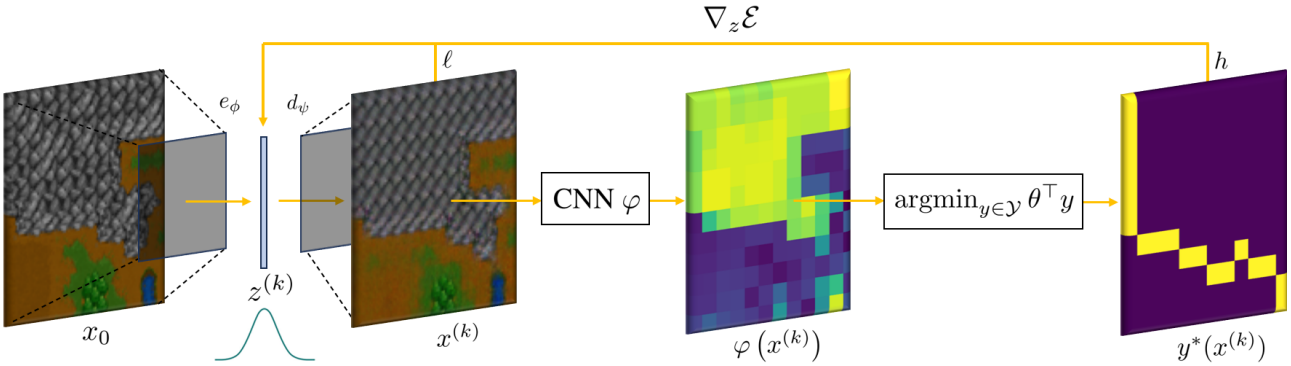


Figure 4. Pipeline of CF-OPT for plausible explanations in high-dimensional spaces. The input x is encoded and decoded using a CA-VAE. The reconstructed context $\tilde{x}$ is given to the prediction model φ, a CNN in our guiding example, to obtain the parameters θ. The parameterized optimization model is finally solved to obtain the decision y^*.

Background. VAEs are probabilistic generative models made of two components. First, an encoder e_ϕ parameterized by ϕ maps any input $x \in \mathbb{R}^{n_x}$ to a distribution $e_\phi(\mathbf{z}|x)$ over the latent space $\mathcal{Z} \subseteq \mathbb{R}^{n_z}$, with $n_z \ll n_x$. Then, a decoder d_ψ parameterized by ψ maps any latent variable $z \in \mathbb{R}^{n_z}$ back to the feature space in the form of a distribution $d_\psi(\mathbf{x}|z)$. The interest of a VAE is to provide a low-dimensional approximation of the unknown manifold underlying the data. The encoder e_ϕ allows for representing x in a much lower dimensional latent variable z, and the decoder d_ψ tries to reconstruct x from z. We will note thereafter $e_\phi(x) = \mathbb{E}\left[e_\phi(\mathbf{z}|x)\right]$ and $d_\psi(z) = \mathbb{E}\left[d_\psi(\mathbf{x}|z)\right]$.

Training a VAE is usually done by maximizing the following lower bound on its log-likelihood, called ELBO (for Evidence Lower Bound, Kingma & Welling (2014)):

$$\begin{aligned}\mathcal{L}(\phi, \psi; x_i) = \mathbb{E}_{z \sim e_\phi(\mathbf{z}|x_i)}\left[\log d_\psi(x_i|z)\right] \\ - \mathrm{D}_{\mathrm{KL}}\left(e_\phi(\mathbf{z}|x_i)||p(\mathbf{z})\right)\end{aligned} \tag{3}$$

where x_i is a sample from the training set, D_{KL} is the Kullback-Leibler (KL) divergence, and $p(\mathbf{z})$ is a prior distribution over the latent variable. As is often done, we set the latter to be an isotropic, centered, multivariate Gaussian distribution: $p(\mathbf{z}) = \mathcal{N}\left(\mathbf{z};\, 0,\, \mathbf{I}_{n_z}\right)$.

3.1. Latent Space Counterfactual Search

The VAE, once trained on the same dataset φ was trained on, allows us to benefit from an approximation of the data manifold by mapping its latent space $\mathcal{Z}$ back to feature space via d_ψ. Thus, we reformulate the plausibility constraint $x \in \mathcal{D}$ of problem (2) in latent space, and state that a close and plausible —latent— counterfactual explanation z^{alt} is a solution to the problem:

$$\min_{z \in \mathcal{Z}} \quad \ell\left(x_0, d_\psi(z)\right) \tag{4a}$$

$$\text{s.t.} \quad \chi(z) = 0, \tag{4b}$$

$$z \in \mathcal{D}_{\mathcal{Z}} \tag{4c}$$

where $\chi(z) = h \circ d_\psi(z)$, and $\mathcal{D}_{\mathcal{Z}}$ is a region of plausibility in latent space. We recover an explanation by taking $x^{\text{alt}} = d_\psi(z^{\text{alt}})$.

Remark 3.1. In several applications, the feature space $\mathcal{X}$ might not be endowed with a meaningful metric to measure the proximity of a tentative counterfactual x^{alt} w.r.t x_0. A convenient substitute is then to measure this distance in the latent space, and use $\ell(x_0, z^{\text{alt}}) = ||e_\phi(x_0) - z^{\text{alt}}||_2^2$ as objective (Deudon, 2018).

3.2. $\mathcal{D}_{\mathcal{Z}}$ as a Latent Hypersphere

It turns out that defining a region of plausibility is easier in the latent space. Indeed, we suggest using $\mathcal{D}_{\mathcal{Z}}$ as the thickened hypersphere

$$\mathcal{D}_{\mathcal{Z}} = \left\{z \in \mathcal{Z} \colon \|z\|_2 \in \left[C_{n_z} - \kappa, C_{n_z} + \kappa\right]\right\}$$

with $C_{n_z} = \mathbb{E}_{z \sim p(\mathbf{z})}\, ||z||_2 \approx \sqrt{n_z}$ and $\kappa > 0$ a small constant. Practically, to ease computations and avoid potential infeasible problems, we use a soft version of the constraint $z \in \mathcal{D}_{\mathcal{Z}}$, leading to the counterfactual explanation problem

$$\min_{z \in \mathcal{Z}} \quad \ell\left(x_0, d_\psi(z)\right) + \Omega(z) \tag{5a}$$

$$\text{s.t.} \quad \chi(z) = 0, \tag{5b}$$

with $\Omega(z) = \beta\left(||z||_2 - C_{n_z}\right)^2$ being our proposed plausibility regularization, and $\beta > 0$ a hyperparameter controlling its weight.

To understand our definition of $\mathcal{D}_{\mathcal{Z}}$, recall that since the KL divergence in Equation (3) forces the distribution of latent codes to match that of the prior $p(\mathbf{z})$, a standard Gaussian is a good approximation of the push-forward of $\mathbb{P}_{\text{data}}$ through the encoder e_ϕ. In Appendix A.1, we provide numerical justifications for this approximation. The Gaussian being isotropic, it is natural to define $\mathcal{D}_{\mathcal{Z}}$ as $\{z \in \mathcal{Z} \colon a \leq ||z||_2 \leq b\}$. Besides, it is well known that the ℓ^2 norm of a high-dimensional standard Gaussian concentrates around its expectation (Biau & Mason, 2015), leading to our definition of $\mathcal{D}_{\mathcal{Z}}$.

In Appendix A.2, we give theoretical insights into why this region is optimal w.r.t the desirable properties we identified in Section 2.2. Moreover, as a benchmark, we experimentally show in Section 5.3 that using this hypersphere instead of a centered ball leads to improved performance (the corresponding region $\tilde{\mathcal{D}}_{\mathcal{Z}}$ being of the form $\{z \in \mathcal{Z} \colon p(z) \geq \delta\}$ and the corresponding regularization being $\tilde{\Omega}(z) = \beta ||z||_2^2$).

3.3. Tailoring VAE Training with a Cost-Aware Loss

The costs predicted after VAE-reconstruction $\varphi(\tilde{x})$ can be quite far from $\varphi(x)$, as training a VAE to maximize the traditional ELBO is agnostic to the downstream task in our specific setting. This is unfortunate since a poor accuracy of the reconstructed costs reduces the quality of the obtained counterfactuals: indeed, it can lead to perturbations in the pipeline's output associated to the initial context x_0.

We adopt an empirical approach to this issue and penalize the distance between predicted costs before and after reconstruction during training. More precisely, we suggest using the following term for sample x_i in the maximized objective when training our VAE in a cost-aware fashion:

$$\begin{aligned}\mathcal{L}_{\text{CA}}(\phi, \psi; x_i) &= \mathcal{L}(\phi, \psi; x_i) \\ &\quad -\alpha\, \mathbb{E}_{z \sim e_\phi(\mathbf{z}|x_i)} \left[||\varphi(x_i) - \varphi\left(d_\psi(z)\right)||_2^2 \right],\end{aligned} \tag{6}$$

where $\mathcal{L}(\phi, \psi; x_i)$ is the ELBO traditionally used, and $\alpha > 0$ is a hyperparameter controlling the regularization weight. Thus, this learning objective takes into account the *cost-reconstruction error*, in addition to the traditional *feature-reconstruction error* (the expectation term in Equation (3)). During training, this additional term is approximated using a Monte-Carlo estimate, as is already usually done when approximating the feature-reconstruction loss (naturally, we use the same samples to compute both).

Remark 3.2. We did not observe a significant impact of the dimension of the latent space on the method, other than its known impact on the ability of the VAE to generate high-quality samples. Hence, the architectural choices follow the usual concerns when training a VAE.

4. Computing Counterfactuals

In general, the explanation problem, whether formulated in feature space or latent space, is non-convex and has no closed-form solution. Our algorithm to obtain explanations is based on constrained differential optimization and, in particular, the modified differential method of multipliers (MDMM) of Platt & Barr (1987).

Similarly to most works on end-to-end training, we also need to propagate a gradient through a discrete optimization layer. However, there is an important difference: the pipeline is already trained. Thus, we compute a gradient w.r.t the covariate's features and not the model's parameters as is done when training the pipeline.

4.1. Augmented Lagrangian and its Gradient

The MDMM is based on the following relaxation of the constrained optimization problem 5, called an *energy function* by Platt & Barr (1987), and commonly known as the augmented Lagrangian (Boyd et al., 2011):

$$\mathcal{E}(z, \lambda) = \ell\left(x_0, d_\psi(z)\right) + \Omega(z) + \lambda\, \chi\left(z\right) + \frac{\rho}{2}\left(\chi\left(z\right)\right)^2 \tag{7}$$

where $\lambda \in \mathbb{R}$ is the dual variable of the explanation constraint and $\rho > 0$ is a constant hyperparameter, called the damping term.

The idea of the MDMM is to perform a gradient descent on $\mathcal{E}$ w.r.t z and a gradient ascent on $\mathcal{E}$ w.r.t λ in order to reach a local minimum z^* that satisfies the constraint $\chi(z^*) = 0$.

Remark 4.1. The primal variable z is initialized at $z^{(1)} = e_\phi(x_0)$, which is the only time the encoder e_ϕ is used per explanation task.

This algorithm is closely related to another first-order constrained optimization algorithm: the method of multipliers (MoM) (Boyd et al., 2011). The main difference is that, at each iteration, the optimization problem is solved to optimality instead of updating z with a gradient descent step. The MDMM is more convenient in our setting since solving $\min_z \mathcal{E}(z, \lambda)$ for each λ is difficult.

As it involves the composition of complex deep architectures (the decoder and the prediction model), the optimization problem considered is highly non-convex. The MDMM can therefore only provide convergence to a local minimum, under rather technical assumptions on its initialization which need not be detailed here (Platt & Barr, 1987). However, our numerical experiments highlight that it is very efficient at obtaining feasible and close solutions.

Gradients. We note $x = d_\psi(z)$. The energy function $\mathcal{E}$ is differentiable almost everywhere w.r.t z (everywhere in the case of relative explanations, the problem coming from differentiating $y^*(x)$ when $\operatorname{argmin}_{y \in \mathcal{Y}} \varphi(x)^\top y$ contains more than one element). Moreover, we have:

$$\nabla \chi(z) = \begin{cases} J_{\varphi \circ d_\psi}(z)\left(y^{\text{alt}} - y_0\right) & \text{(relative exp.)} \\ J_{\varphi \circ d_\psi}(z)\left(y^{\text{alt}} - y^*(x)\right) & \text{(absolute exp.)} \\ J_{\varphi \circ d_\psi}(z)\left((1+\varepsilon)\, y^*(x) - y_0\right) & (\varepsilon\text{-exp.}) \end{cases} \tag{8}$$

These gradients can be derived easily since $\nabla_x y^*(x) = 0$ almost everywhere due to y^* being piecewise constant, as the solution of a linear optimization problem. The Jacobian $J_{\varphi \circ d_\psi}(z)$ is computed via automatic differentiation through decoder d_ψ and prediction model φ.

Remark 4.2. If $\min_{y \in \mathcal{Y}} \varphi(x)^\top y$ is negative, we only have to change $(1+\varepsilon)$ to $(1-\varepsilon)$ inside the expression of the gradient for ε-explanations (see Remark 2.4).

4.2. Our Algorithm

Our algorithm is given in Algorithm 1 in Appendix D. The parameters K and $c_{\max}$ are hyperparameters that define the maximum number of iterations and the maximum number of non-improving iterations. We define an iteration k as improving if $x^{(k)}$ satisfies the explanation criterion, and if $\ell(x_0, x^{(k)}) + \Omega(z^{(k)}) \leq u \left(\ell(x_0, x^{(j)}) + \Omega(z^{(j)})\right)$, where j is the index of the last improving iteration and $u \in]0, 1[$ is an update tolerance hyperparameter.

If the plausibility of the explanation is not taken into account, the use of a VAE is not necessary and the feature space formulation of `CF-OPT` is given in Algorithm 2.

The computational effort of finding an explanation of a decision lies in computing the gradient in Equation (8). For absolute and ε-explanations, the linear optimization problem $\min_{y \in \mathcal{Y}} \varphi(x^{(k)})^\top y$ is solved once per iteration. The solution $y^*(x^{(k)})$ is used twice at each iteration: to check whether the explanation criterion is satisfied and to compute the gradient $\nabla_z \mathcal{E}$. Relative explanations do not require to solve this problem and are thus cheaper to compute.

5. Numerical Study

In this section, we evaluate the ability of `CF-OPT` to obtain plausible explanations of decisions in a high-dimensional setting. We focus on the problem of finding shortest paths on Warcraft maps, which has been our guiding example throughout the paper. Warcraft maps are 96×96 RGB images with continuous color values. Every area is walkable, and crossing them results in different costs. The true, unknown costs can only take discrete values (associated with the terrain being a forest, rock, water, etc.), but the Convolutional Neural Network (CNN) φ used to predict them from the maps outputs continuous ones. The output of the full structured learning pipeline is the shortest path from the upper-left corner to the lower-right one.

Our experiments study the value of using the hypersphere regularization and training a VAE in a cost-aware fashion. We provide the details of our experimental setting, the architecture of our models, as well as other results with low-dimensional tabular data in Appendix B.

Our experiments are implemented in Python and run on four cores of an Intel Core i7-8565U CPU @ 1.80GHz and use 16GB RAM. The code used to generate all the results in this paper is available publicly at `https://github.com/GermainVivierArdisson/CF-OPT` under an MIT license.

5.1. Experimental Setting

We follow the experimental setting of Vlastelica et al. (2019) and used subsequently in several works (Dalle et al., 2022; Tang & Khalil, 2022; McKenzie et al., 2023). The training set $\{x_i, \theta_i, y_i\}_{i=1}^N$ is made of $N = 10000$ examples of Warcraft maps as well as their associated true costs and shortest paths.

Relative regret metric. We introduce here the relative regret metric, which is used in our experiments to quantify the impact of both the cost-aware and plausibility regularization. The relative regret $\mathrm{reg}(y, \theta)$ of a feasible solution $y \in \mathcal{Y}$ for a cost θ is defined as:

$$\mathrm{reg}(y, \theta) = \frac{\theta^\top y - \theta^\top y^*(\theta)}{|\theta^\top y^*(\theta)|}, \tag{9}$$

where θ is a cost vector and and $y^*(\theta) = \mathrm{argmin}_{y \in \mathcal{Y}} \theta^\top y$ is the optimal solution associated to θ.

VAE models. All VAEs are implemented as CNNs with convolutions for the encoder and transposed convolutions for the decoder. The default latent space dimension is taken to be 64.

Selection of explanation tasks. In our experiments, we repeatedly solve explanation problems using initial maps and alternative paths to obtain N_{exp} explanations. The initial maps are all different and never seen during the training of any VAE. Further, to avoid trivial explanation tasks, the alternative paths defining explanation problems are always taken to be different from the shortest path associated to the initial map.

5.2. Impact of Cost-Aware Training

First, we quantify the impact of the cost-aware regularization in the training objective of the VAE in Equation (6) by measuring the reconstruction of the optimal path after the VAE. We measure this with three metrics:

(i) the relative regret $\mathrm{reg}\left(y^*(\varphi(x_i)),\, \varphi(\tilde{x}_i)\right)$ of the initial optimal solution associated to x_i in the reconstructed context $\tilde{x}_i$,

(ii) the relative regret $\mathrm{reg}\left(y^*(\theta_i),\, \varphi(\tilde{x}_i)\right)$ of the optimal path associated to the true cost θ_i in the reconstructed context $\tilde{x}_i$,

(iii) the squared ℓ^2 norm $||x_i - \tilde{x}_i||_2^2$, the natural metric for evaluating the feature-reconstruction performances of the VAE.

We train several VAEs, with varying cost-aware regularization weight α. Notice that using $\alpha = 0$ recovers the traditional cost-agnostic VAE. All VAEs are trained until convergence, with early stopping to avoid overfitting. After training, we measure the average of each performance metric over a test set of 1000 data points.

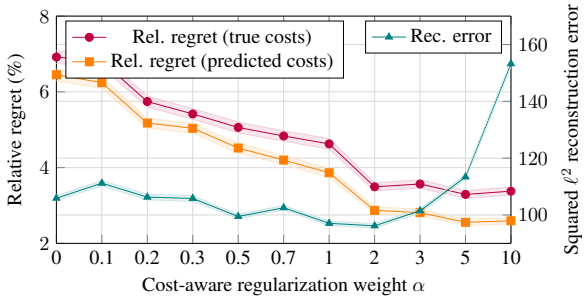


Figure 5. Comparison of VAE and Cost-Aware VAE for varying α

The results in Figure 5 show that the relative regret metrics defined above clearly decrease when increasing the regularization weight. Hence, introducing a cost-aware regularization is very powerful to decrease cost-reconstruction error, and to stabilize the optimal paths associated with the Warcraft maps under VAE-reconstruction. As expected, the relative regret is lower when taking the initial optimal path associated with the costs predicted by φ rather than the true ones. Figure 5 further shows that the reconstruction error decreases for $0 < \alpha \leq 2$. This result suggests that increasing the regularization weight even guides the training process and helps the model to reconstruct images. Consequently, we set the cost-regularization parameter to $\alpha = 2$ for the rest of the experiments.

To further highlight the impact of the cost-aware regularization w.r.t the explanation problem, we analyze the loss of obtained explanations when using trained Cost-Aware VAEs with different regularization weights. For each weight, we run `CF-OPT` on $N_{\text{exp}} = 50$ different absolute explanation tasks. We use a step size $\gamma = 0.003$, maximum number of iterations $K = 3000$, maximum number of non-improving iterations $c_{\max} = 50$, and corresponding update tolerance $u = 0.9$. We then measure the loss of the best solution according to our best VAE using latent hypersphere plausibility regularization with $\beta = 10$, and a feature space objective. The loss is thus equal to $||x_{\text{best}} - x_0||_2^2 + 10\,(||z_{\text{best}}||_2 - C_{64})^2$. As seen in Figure 6, Cost-Aware VAEs trained with $\alpha \geq 0.5$ lead to significantly better counterfactual explanations in terms of the resulting loss.

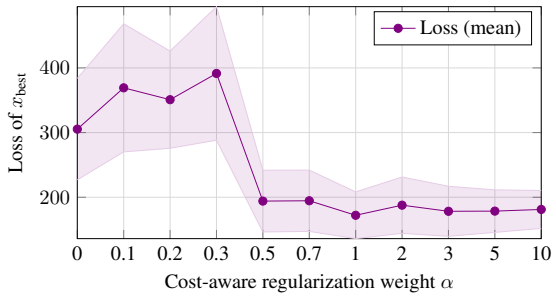


Figure 6. Loss of the obtained explanations when using Cost-Aware VAE with varying α

5.3. Impact of Plausibility Regularization

We now analyze the impact of the proposed latent hypersphere regularization on the plausibility of the obtained counterfactual explanations. Instead of relying only on subjective visual comparisons of the results, we use two different metrics to quantify plausibility.

Reconstruction error. First, we use a well-known metric in outlier detection (An & Cho, 2015) measuring the reconstruction error of a VAE trained on the corresponding dataset when evaluated on the potential outlier. We select $N_{\text{exp}} = 40$ absolute explanation tasks as above and run `CF-OPT` with the same hyperparameters but varying the weight β of the latent hypersphere regularization. Then, we measure the reconstruction error with a Cost-Aware VAE trained on the Warcraft maps dataset with $\alpha = 2$, when evaluated on the returned best solutions x_{best}. The results are displayed in Figure 7 and show that the proposed latent hypersphere regularization leads to more plausible explanations w.r.t this outlier detection metric.

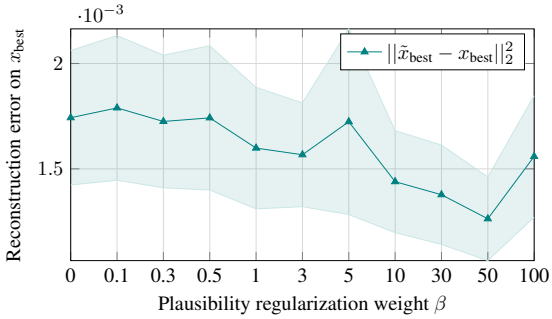
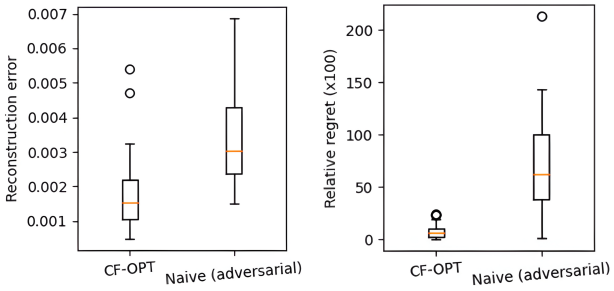


Figure 7. Reconstruction error of a Cost-Aware VAE ($\alpha = 2$) on the best solution obtained x_{best}, with varying latent hypersphere regularization weight β.

Decision-focused reconstruction error. Our second metric is specifically designed for our setting. Instead of simply measuring the reconstruction error in the feature space as above, we measure the stability of the optimal path associated to the returned best solution x_{best} under reconstruction with a Cost-Aware VAE. This metric is more suited to detect adversarial explanations since adversarial perturbations tend to simply vanish when passing them through a VAE, due to the information compression aspect of the model (which then acts as a denoising model), thus leading to a small reconstruction error. To do so, we measure the relative regret $\text{reg}\left(y^*(\varphi(x_{\text{best}})),\ \varphi(\tilde{x}_{\text{best}})\right)$ of the shortest path associated with x_{best} in the reconstructed context $\tilde{x}_{\text{best}}$ when using a Cost-Aware VAE trained with $\alpha = 2$. This metric extends the idea of reconstruction error to the entire data-driven optimization pipeline.

We obtain $N_{\text{exp}} = 40$ absolute explanations using a latent-space search with `CF-OPT`, and $N_{\text{exp}} = 40$ other counterfactuals using a naive search (i.e., adversarial explanations



(a) Reconstruction error (b) Decision-focused reconstruction error

Figure 8. (a) Task-agnostic reconstruction error and (b) decision-focused reconstruction error for detecting adversarial explanations.

as in Figure 3). For CF-OPT, the VAE used for optimization is not cost-regularized ($\alpha = 0$) in order to avoid biases since the proposed metric is computed using a Cost-Aware VAE itself. As shown in Figure 8, our decision-focused metric is more efficient at differentiating between adversarial and plausible explanations than the classical reconstruction error.

Hence, we use this metric to measure the plausibility of explanations obtained with CF-OPT and latent hypersphere regularization for varying regularization weight β. Notice again that $\beta = 0$ recovers the unregularized explanation objective. Similarly to the experiment shown in Figure 8, the decision-focused reconstruction error is computed on $N_{\text{exp}} = 40$ absolute explanation tasks. Our results in Figure 9 show that the latent hypersphere regularization also improves the reconstruction of the pipeline's output associated with the produced counterfactuals.

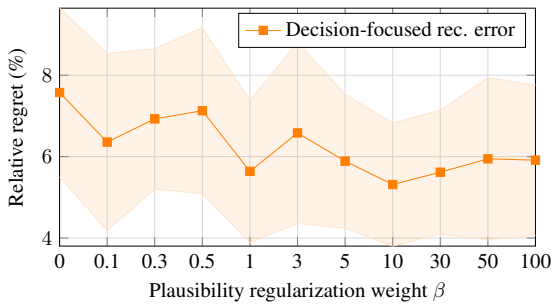


Figure 9. Decision-focused reconstruction error measured with an $\alpha = 2$ Cost-Aware VAE, for varying latent hypersphere regularization weight β.

Comparison of latent hypersphere and log-likelihood regularizations. We now investigate the value of the proposed latent hypersphere regularization Ω, compared to the log-likelihood regularization $\tilde{\Omega}$ (both introduced in Section 3.2). We select $N_{\text{exp}} = 40$ absolute explanation tasks and run CF-OPT on each explanation task with a Cost-Aware VAE trained with $\alpha = 2$. We measure the loss of the best solution found with latent hypersphere regularization and with log-likelihood regularization for varying regularization weight β.

The proximity metric used is the squared euclidean distance in feature space, so that the corresponding loss is $||x_{\text{best}} - x_0||_2^2 + \beta\left(||z_{\text{best}}||_2 - C_{64}\right)^2$ for the latent hypersphere regularization, and $||x_{\text{best}} - x_0||_2^2 + \beta||z_{\text{best}}||_2^2$ for the log-likelihood regularization. As can be seen in Figure 10, the latent hypersphere regularization does not significantly impact the loss of the obtained counterfactual, contrary to the log-likelihood regularization which leads to an exponential increase in the loss as β increases.

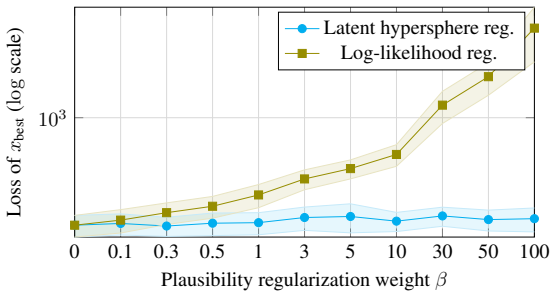


Figure 10. Loss of the obtained explanations with latent hypersphere (Ω) and log-likelihood ($\tilde{\Omega}$) regularizations, with varying weight β.

6. Related Literature

This work lies at the intersection of end-to-end learning, generative models, and explainable machine learning. We briefly outline some relevant literature.

End-to-end learning and optimization. Integrating optimization into learning pipelines has received a lot of attention. The main challenge is to train the pipeline in an end-to-end fashion, that is, to propagate the loss associated to the objective function of the pipeline's optimization component back to its predictive component. This requires differentiating through the pipeline, particularly through the argmin operation of the optimization component. Several approaches have been proposed based, for instance, on implicit differentiation (Amos & Kolter, 2017; Donti et al., 2017), using a convex surrogate loss (Elmachtoub & Grigas, 2022), or perturbed optimizers (Berthet et al., 2020; Dalle et al., 2022). We refer to the survey of Mandi et al. (2023) and Sadana et al. (2023) for further details.

Counterfactual explanations. Counterfactual explanations have received significant attention to explain learning models (Wachter et al., 2017; Verma et al., 2020), especially in the classification setting, where they translate as the smallest alteration in the input features that changes the output's label into another, predefined one.

Generative models and plausible explanations. Generative models are also used to provide counterfactual explanations that adhere to the data manifold in the predictive setting, that is when the model to be explained is a classifier. Pawelczyk et al. (2020) use a modified version of the VAE, dubbed C-CHVAE, and randomly sample latent codes on a sphere centered at the latent embedding of the initial context until the explanation condition is met. This would be very algorithmically inefficient in our setting because of the combinatorial aspect of the problem: there tends to be an exponential number of (feasible) decisions even for simpler linear problems such as the shortest path, compared to the reasonable number of classes in the predictive setting. This also prevents using methods that train separate VAE models for each class, such as the one presented by Barr et al. (2021). Liu et al. (2019) minimize a relaxed objective inside the latent space of a trained generative adversarial network (GAN) using gradient descent. Since GANs are implicit likelihood models, one can not derive a regularization on the plausibility of the generated counterfactual as easily as we did with a VAE in Section 3.2.

7. Conclusions

This paper presented an approach to obtain counterfactual explanations of the decisions of end-to-end optimization pipelines. It focuses on obtaining plausible explanations in high-dimensional settings such as when the input are images. One limitation of the current approach is that the covariates x must be made of continuous features. Extending the work to categorical or discrete features is a promising research direction. A central component of the method is the use of a VAE, which are especially relevant in our setting because of the smaller dimension of the latent space. Another interesting direction for future work is to investigate the use of other generative models such as diffusion models.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. It highlights the vulnerability of structured learning pipelines to adversarial examples. It then provides an efficient algorithm to obtain plausible explanations. This has several potential beneficial impacts. Explanations can improve the transparency of these pipelines, build trust in their decisions, and highlight potential biases.

References

Amos, B. and Kolter, J. Z. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pp. 136–145. PMLR, 2017.

An, J. and Cho, S. Variational autoencoder based anomaly detection using reconstruction probability. *Special lecture on IE*, 2(1):1–18, 2015.

Barr, B., Harrington, M. R., Sharpe, S., and Bruss, C. B. Counterfactual explanations via latent space projection and interpolation. *arXiv preprint arXiv:2112.00890*, 2021.

Berthet, Q., Blondel, M., Teboul, O., Cuturi, M., Vert, J.-P., and Bach, F. Learning with differentiable perturbed optimizers. *Advances in Neural Information Processing Systems*, 33:9508–9519, 2020.

Biau, G. and Mason, D. M. *High-Dimensional p-Norms*, pp. 21–40. Springer International Publishing, Cham, 2015.

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

Browne, K. and Swift, B. Semantics and explanation: why counterfactual explanations produce adversarial examples in deep neural networks. *arXiv preprint arXiv:2012.10076*, 2020.

Dalle, G., Baty, L., Bouvier, L., and Parmentier, A. Learning with combinatorial optimization layers: a probabilistic approach. *arXiv preprint arXiv:2207.13513*, 2022.

Deudon, M. Learning semantic similarity in a continuous space. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31, 2018.

Donti, P., Amos, B., and Kolter, J. Z. Task-based end-to-end model learning in stochastic optimization. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Elmachtoub, A. N. and Grigas, P. Smart "predict, then optimize". *Management Science*, 68(1):9–26, 2022.

Elmachtoub, A. N., Liang, J. C. N., and McNellis, R. Decision trees for decision-making under the predict-then-optimize framework. In *International Conference on Machine Learning*, pp. 2858–2867. PMLR, 2020.

Ferber, A., Wilder, B., Dilkina, B., and Tambe, M. MIPaaL: Mixed integer program as a layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, no. 02, pp. 1504–1511, 2020.

Ferber, A. M., Huang, T., Zha, D., Schubert, M., Steiner, B., Dilkina, B., and Tian, Y. SurCo: Learning linear SURrogates for COmbinatorial nonlinear optimization problems. In *International Conference on Machine Learning*, pp. 10034–10052. PMLR, 2023.

Forel, A., Parmentier, A., and Vidal, T. Explainable data-driven optimization: From context to decision and back again. In *International Conference on Machine Learning*, pp. 10170–10187. PMLR, 2023.

Freiesleben, T. The intriguing relation between counterfactual explanations and adversarial examples. *Minds and Machines*, 32(1):77–109, 2022.

Gupta, R. and Zhang, Q. Data-driven decision-focused surrogate modeling. *AIChE Journal*, pp. e18338, 2024.

Jeong, J., Jaggi, P., Butler, A., and Sanner, S. An exact symbolic reduction of linear smart predict+ optimize to mixed integer linear programming. In *International Conference on Machine Learning*, pp. 10053–10067. PMLR, 2022.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *International Conference for Learning Representations*, 2015.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2014.

Liu, S., Kailkhura, B., Loveland, D., and Han, Y. Generative counterfactual introspection for explainable deep learning. In *2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 1–5, 2019.

Ma, X., Niu, Y., Gu, L., Wang, Y., Zhao, Y., Bailey, J., and Lu, F. Understanding adversarial attacks on deep learning based medical image analysis systems. *Pattern Recognition*, 110:107332, 2021.

Mandi, J., Kotary, J., Berden, S., Mulamba, M., Bucarey, V., Guns, T., and Fioretto, F. Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. *arXiv preprint arXiv:2307.13565*, 2023.

McKenzie, D., Fung, S. W., and Heaton, H. Faster predict-and-optimize with three-operator splitting. *arXiv preprint arXiv:2301.13395*, 2023.

Niepert, M., Minervini, P., and Franceschi, L. Implicit MLE: backpropagating through discrete exponential family distributions. *Advances in Neural Information Processing Systems*, 34:14567–14579, 2021.

Odaibo, S. Tutorial: Deriving the standard variational autoencoder (VAE) loss function. *arXiv preprint arXiv:1907.08956*, 2019.

Pawelczyk, M., Haug, J., Broelemann, K., and Kasneci, G. Learning model-agnostic counterfactual explanations for tabular data. In *Proceedings of The Web Conference 2020*, pp. 3126–3132, April 2020.

Pawelczyk, M., Agarwal, C., Joshi, S., Upadhyay, S., and Lakkaraju, H. Exploring counterfactual explanations through the lens of adversarial examples: A theoretical and empirical analysis. In *International Conference on Artificial Intelligence and Statistics*, pp. 4574–4594. PMLR, 2022.

Platt, J. and Barr, A. Constrained differential optimization. *Advances in Neural Information Processing Systems*, 1987.

Sadana, U., Chenreddy, A., Delage, E., Forel, A., Frejinger, E., and Vidal, T. A survey of contextual optimization methods for decision making under uncertainty. *arXiv preprint arXiv:2306.10374*, 2023.

Sahoo, S. S., Paulus, A., Vlastelica, M., Musil, V., Kuleshov, V., and Martius, G. Backpropagation through combinatorial algorithms: Identity with projection works. In *International Conference on Learning Representations*, 2023.

Stewart, L., Bach, F. S., López, F. L., and Berthet, Q. Differentiable clustering with perturbed spanning forests. In *Advances in Neural Information Processing Systems*, 2023.

Tang, B. and Khalil, E. B. Pyepo: A pytorch-based end-to-end predict-then-optimize library for linear and integer programming. *arXiv preprint arXiv:2206.14234*, 2022.

Verma, S., Dickerson, J., and Hines, K. Counterfactual explanations for machine learning: A review. *arXiv preprint arXiv:2010.10596*, 2020.

Vlastelica, M., Paulus, A., Musil, V., Martius, G., and Rolinek, M. Differentiation of blackbox combinatorial solvers. In *International Conference on Learning Representations*, 2019.

Wachter, S., Mittelstadt, B., and Russell, C. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harvard Journal of Law & Technology*, 31:841, 2017.

Wilder, B., Dilkina, B., and Tambe, M. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):1658–1665, 2019.

A. Numerical and Theoretical Justifications for the Plausibility Regularization

A.1. Numerical Validation of the Approximation of the Encoded Data Distribution by the Prior

An important assumption we make in order to derive the plausibility region $\mathcal{D}_{\mathcal{Z}}$ and the corresponding plausibility regularization Ω in Section 3.2 is that the prior over the latent codes $p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I}_{n_z})$ is a good approximation of the true encoded data distribution, which can be thought of as the push-forward of $\mathbb{P}_{\text{data}}$ through the trained encoder e_ϕ of the VAE.

This approximation is motivated by the fact that the KL divergence in the training objective of the VAE forces the distribution of the latent codes to match that of the prior $p(\mathbf{z})$. This same approximation is made when using trained VAEs for traditional generative modeling, as the prior is used to sample new data points before passing them through the trained decoder d_ψ.

To verify quantitatively if this assumption holds, we compare the percentage of points that should belong to the plausibility region according to the prior (that is, the mass of the region w.r.t $p(\mathbf{z})$) and the percentage of training data points that are actually embedded in the region (that is, the mass of the region w.r.t the empirical encoded data distribution). The table below shows the percentage of plausible points for varying thickness κ (in Euclidean distance) of the thickened hypersphere of radius C_{64} when using a VAE with 64-dimensional latent space.

Table 1. Comparison of prior and empirical encoded data distribution

	$\kappa = 0.0$	$\kappa = 0.25$	$\kappa = 0.5$	$\kappa = 0.75$	$\kappa = 1.0$	$\kappa = 1.25$	$\kappa = 1.5$	$\kappa = 1.75$	$\kappa = 2.0$
Prior (%)	0.0	27.6	52.1	71.2	84.4	92.4	96.7	98.7	99.5
Empirical (%)	0.0	27.8	52.1	71.2	84.8	93.0	97.0	99.1	99.8

Thus, the empirical percentages of training data points embedded in the chosen plausibility region closely match those computed with the statistics of the prior, which further justifies its use as an approximation of the encoded data distribution.

A.2. Latent Hypersphere as Optimal Plausibility Region

Let $p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I}_{n_z})$ be the prior over the latent codes. We use it to approximate the true encoded data distribution, as explained and numerically justified in Appendix A.1. We want to find a plausibility region $\mathcal{D}_{\mathcal{Z}} \subseteq \mathcal{Z} = \mathbb{R}^{n_z}$ with minimal (i) expected distance to latent codes, and (ii) volume, or Lebesgue measure.

These two objectives are conflicting: indeed, the optimal region with respect to the first one only is the whole space, and the optimal region with respect to the second one only is any region that is negligible with respect to the Lebesgue measure. Thus, they cannot be jointly minimized, so we consider the following problem:

$$\mathcal{D}_{\mathcal{Z}} = \underset{\mathcal{D} \subseteq \mathcal{Z}}{\operatorname{argmin}} \quad \mathbb{E}_{z \sim p(\mathbf{z})} \left[\ell(z, \mathcal{D}) \right] + \eta \, \text{Leb}(\mathcal{D}),$$

where $\eta > 0$ is a trade-off coefficient, and $\ell(z, \mathcal{D}) = \min_{z' \in \mathcal{D}} \{ \|z - z'\|_2^2 \}$ is the squared Euclidean distance of latent vector z to region $\mathcal{D}$.

We assume $\mathcal{D}_{\mathcal{Z}}$ to be an isotropic region to respect the symmetries of the encoded data distribution (which is approximated by the isotropic centered Gaussian $p(\mathbf{z})$). Thus, one can parameterize it as a union of spheres $\mathcal{D}_{\mathcal{Z}} = \left\{ z \in \mathcal{Z} \colon \|z\|_2 \in R \right\}$, where $R \subseteq \mathbb{R}_+$ is the set of radii of the spheres. If we further assume $\mathcal{D}_{\mathcal{Z}}$ to be connected, which appears logical as $p(\mathbf{z})$ is unimodal, one has that R is an interval of $\mathbb{R}_+$, which we consider to be closed. These assumptions allow us to consider the following simplified optimization objective:

$$\begin{aligned} \underset{a,\, b \in \mathbb{R}_+}{\operatorname{argmin}} \quad & \mathbb{E}_{z \sim p(\mathbf{z})} \left[\ell(z, \mathcal{D}_a^b) \right] + \eta \, \text{Leb}(\mathcal{D}_a^b) \\ \text{s.t.} \quad & a \leq b, \end{aligned} \tag{10}$$

where $\mathcal{D}_a^b = \left\{ z \in \mathcal{Z} \colon \|z\|_2 \in [a, b] \right\}$. Thus, the only candidate regions left are either centered balls (when $a = 0$), centered hyperspheres (when $a = b$), and thickened centered hyperspheres (when $0 < a < b$).

For η large enough in Problem (10), the weight of the second term of the optimization objective forces the optimal region to have approximately zero Lebesgue volume. Thus, we can approximate it as $\mathcal{D}_a^a$ (that is, to be a centered, non-thickened

hypersphere of radius a), and the problem can be approximated as:

$$\underset{a\in\mathbb{R}_+}{\operatorname{argmin}} \quad \mathbb{E}_{z\sim p(\mathbf{z})}(a-||z||_2)^2, \tag{11}$$

which is equivalent to:

$$\underset{a\in\mathbb{R}_+}{\operatorname{argmin}} \quad \mathbb{E}_{X\sim\chi_{n_z}}(a-X)^2,$$

where χ_{n_z} is a Chi distribution with n_z degrees of freedom since $p(\mathbf{z})$ is a standard n_z-dimensional Gaussian. Thus, the solution is the expectation of the χ_{n_z} distribution, $a = C_{n_z} = \sqrt{2}\frac{\Gamma\left(\frac{n_z+1}{2}\right)}{\Gamma\left(\frac{n_z}{2}\right)} \approx \sqrt{n_z}$, and we find the **centered hypersphere of radius** C_{n_z} as optimal plausibility region.

Numerical valdiation. We investigate the validity of this approximation numerically. On the one hand, we have:

$$\text{Leb}(\mathcal{D}_a^b) = \frac{\pi^{\frac{n_z}{2}}}{\Gamma(n_z/2+1)}(b^{n_z}-a^{n_z}),$$

and on the other hand:

$$\begin{aligned}
\mathbb{E}_{z\sim p(\mathbf{z})}\left[\ell(z,\mathcal{D}_a^b)\right] &= \int_0^a (a-r)^2(2\pi)^{-\frac{n_z}{2}}e^{-\frac{r^2}{2}}S^{n_z-1}r^{n_z-1}dr + \int_b^{+\infty}(r-b)^2(2\pi)^{-\frac{n_z}{2}}e^{-\frac{r^2}{2}}S^{n_z-1}r^{n_z-1}dr \\
&\text{(where } S^{n_z-1} = \frac{2\pi^{\frac{n_z}{2}}}{\Gamma\left(\frac{n_z}{2}\right)} \text{ is the surface of the hypersphere of dimension } n_z-1 \text{ and radius 1)} \\
&= a^2\,\mathbb{P}\left(\mathbf{z}\in B(0,a)\right) + b^2\,\mathbb{P}\left(\mathbf{z}\in B(0,b)^c\right) \\
&- 2a\int_0^a (2\pi)^{-\frac{n_z}{2}}e^{-\frac{r^2}{2}}S^{n_z-1}r^{n_z}dr + \int_0^a (2\pi)^{-\frac{n_z}{2}}e^{-\frac{r^2}{2}}S^{n_z-1}r^{n_z+1}dr \\
&- 2b\int_b^{+\infty} (2\pi)^{-\frac{n_z}{2}}e^{-\frac{r^2}{2}}S^{n_z-1}r^{n_z}dr + \int_b^{+\infty} (2\pi)^{-\frac{n_z}{2}}e^{-\frac{r^2}{2}}S^{n_z-1}r^{n_z+1}dr \\
&= a^2\,P\left(\frac{n_z}{2},\,\frac{a^2}{2}\right) + b^2\,Q\left(\frac{n_z}{2},\,\frac{b^2}{2}\right) \\
&- \frac{2\sqrt{2}}{\Gamma\left(\frac{n_z}{2}\right)}\left[a\left(\Gamma\left(\frac{n_z+1}{2}\right) - \Gamma\left(\frac{n_z+1}{2},\frac{a^2}{2}\right)\right) + b\,\Gamma\left(\frac{n_z+1}{2},\frac{b^2}{2}\right)\right] \\
&+ \frac{2}{\Gamma\left(\frac{n_z}{2}\right)}\left[\Gamma\left(\frac{n_z}{2}+1\right) - \Gamma\left(\frac{n_z}{2}+1,\frac{a^2}{2}\right) + \Gamma\left(\frac{n_z}{2}+1,\frac{b^2}{2}\right)\right],
\end{aligned}$$

where $\Gamma(\cdot\,,\cdot)$ is the upper incomplete gamma function, and $Q(s,x) = \frac{\Gamma(s,x)}{\Gamma(s)} = 1 - P(s,x)$ with $P(s,x)$ being the cumulative distribution function of a gamma random variable with shape parameter s and scale parameter 1.

Putting all the pieces together, we have:

$$\begin{aligned}
\mathbb{E}_{z\sim p(\mathbf{z})}\left[\ell(z,\mathcal{D}_a^b)\right] + \eta\,\text{Leb}(\mathcal{D}_a^b) &= a^2\,P\left(\frac{n_z}{2},\,\frac{a^2}{2}\right) + b^2\,Q\left(\frac{n_z}{2},\,\frac{b^2}{2}\right) \\
&- 2\sqrt{2}\frac{\Gamma\left(\frac{n_z+1}{2}\right)}{\Gamma\left(\frac{n_z}{2}\right)}\left[a\,P\left(\frac{n_z+1}{2},\,\frac{a^2}{2}\right) + b\,Q\left(\frac{n_z+1}{2},\,\frac{b^2}{2}\right)\right] \\
&+ n_z\left[P\left(\frac{n_z}{2}+1,\,\frac{a^2}{2}\right) + Q\left(\frac{n_z}{2}+1,\,\frac{b^2}{2}\right)\right] + \eta\,\frac{\pi^{\frac{n_z}{2}}}{\Gamma(n_z/2+1)}(b^{n_z}-a^{n_z}).
\end{aligned}$$

Using the analytical expressions derived above, we verify that our reasoning holds by solving Problem (10) numerically. To do so, we compute $\Gamma(\cdot)$, $P(\cdot\,,\cdot)$ and $Q(\cdot\,,\cdot)$ using the functions `special.gamma`, `special.gammainc`, and `special.gammaincc`, respectively, taken from the `scipy.special` package. We conduct a simple grid search, by taking 10^4 values for a and b between 0 and 10, discarding couples such that $a > b$. For $n_z = 64$, we find that the hypersphere of radius $C_{64} = \sqrt{2}\frac{\Gamma\left(\frac{65}{2}\right)}{\Gamma\left(\frac{64}{2}\right)} \approx 7.97$ is indeed optimal for $\eta \geq 10^{-16}$.

B. Supplementary Material: Numerical Study

In this appendix, we provide background details on VAEs as well as the architecture of the VAEs used in our experiments in Appendix B.1, and additional experiments on tabular data in Appendix B.2.

B.1. VAE: Background and Detailed Architecture

B.1.1. Background on VAEs

The encoder e_ϕ is chosen to output a Gaussian distribution with a diagonal covariance matrix, that is:

$$e_\phi(\mathbf{z}|x) = \mathcal{N}\left(\mathbf{z};\, \mu_\phi(x),\, \text{Diag}\left[\sigma^2_\phi(x)\right]\right),$$

where $\mu_\phi(x),\ \log \sigma^2_\phi(x) = \log(\sigma^2_\phi(x)) \in \mathbb{R}^{n_z}$ (log being applied element-wise) are the outputs of the encoder neural network for the input $x \in \mathbb{R}^{n_x}$. We choose the decoder to output an isotropic Gaussian distribution, that is:

$$d_\psi(\mathbf{x}|z) = \mathcal{N}(\mathbf{x};\, \mu_\psi(z),\, \mathbf{I}_{n_x})$$

where $\mu_\psi(z) \in \mathbb{R}^{n_x}$ is the output of the decoder neural network for the input $z \in \mathbb{R}^{n_z}$.

Because of our choices for the encoder and decoder, the term inside the expectation in the ELBO 3 is equal to $-\frac{1}{2}||\mathbf{x}_i - \mu_\psi(z)||_2^2$, and the KL divergence has a closed-form expression, so that we have (see Odaibo (2019) for a guided derivation):

$$\mathcal{L}(\phi, \psi, x_i) = -\frac{1}{2}\,\mathbb{E}_{z\sim e_\phi(\mathbf{z}|x_i)}\left[||x_i - d_\psi(z)||_2^2\right] + \frac{1}{2}\sum_{k=1}^{n_z}\left[1 + \log\sigma^2_\phi(x)_k - \sigma_\phi(x)^2_k - \mu_\phi(x)^2_k\right].$$

The expectation in this expression, known as the reconstruction loss (or feature-reconstruction loss to be more precise in our setting), is approximated via a Monte-Carlo estimate during training.

B.1.2. Architecture for the Warcraft Map Experiments

All VAEs used in our experiments are implemented in Pytorch.

The encoder's architecture is the following:

- Convolutional Layers:
  - `conv1`: A 2D convolutional layer with 32 filters, a kernel size of 4x4, a stride of 2, and padding of 1. It processes the input data, of size (Batch-size $\times$ 3 $\times$ 96 $\times$ 96) and produces feature maps.
  - `conv2`: Another 2D convolutional layer with 64 filters, same kernel size, stride, and padding. It further extracts features.
  - `conv3`: A 2D convolutional layer with 128 filters, same parameters as above.
  - `conv4`: The final 2D convolutional layer with 256 filters, again with the same settings.
- Fully Connected Layers:
  - `fc1`: A fully connected layer with 512 output neurons. It processes the flattened feature maps from the last convolutional layer.
  - `fc21` and `fc22`: These fully connected layers produce two vectors: $\mu_\phi(x)$ and $\log\sigma^2_\phi(x)$ (representing the log variance). These vectors are used to sample points in the latent space.
- Activation Functions: we use ReLU activation functions after each layer except after `fc21` and `fc22`.

The decoder's architecture is the following:

- Fully Connected Layers:
  - `fc10`: A fully connected layer with 512 output neurons, that takes the latent vector as input and produces a feature representation.
  - `fc20`: Another fully connected layer that expands the feature representation back to the original flattened shape (6x6x256).
- Transposed Convolutional Layers (also known as deconvolutional layers):

- `deconv1`: A transposed 2D convolutional layer with 128 filters, kernel size 4x4, stride 2, and padding 1. It upsamples the feature maps.
- `deconv2`: Another transposed 2D convolutional layer with 64 filters, same parameters.
- `deconv3`: A transposed 2D convolutional layer with 32 filters, same settings.
- `deconv4`: The final transposed 2D convolutional layer with 3 filters (giving RGB channels), same parameters. It generates the reconstructed data.

- Activation Functions: we use ReLU activation functions after each layer except after `deconv4`, where we use a Sigmoid function.

B.2. Repeated Experiments on Tabular Data

Experimental setting. We perform repeated experiments generating counterfactual explanations of decisions and measure the sensitivity of our method to changes in the problem settings. We measure the total number of iterations of `CF-OPT` when varying the number of features, the number of decision variables, and the complexity of the prediction model.

Remark B.1. We do not measure the sensitivity of our method to the number of training points in the dataset since the computational complexity of calculating an explanation with `CF-OPT` does not increase with the size of the training dataset. This is a clear improvement compared to Forel et al. (2023).

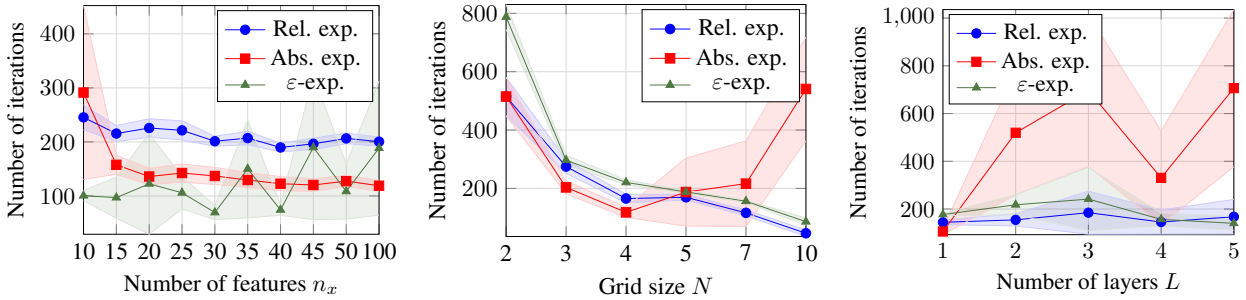
B.2.1. Shortest paths on a grid.

We apply our explanation method to the shortest-path problem on a square grid introduced by Elmachtoub & Grigas (2022) and used in several other works (Elmachtoub et al., 2020; Tang & Khalil, 2022). The structured learning pipeline consists of a contextual vector $x \in \mathbb{R}^{n_x}$, a linear regression model φ predicting costs $\varphi(x) = \theta \in \mathbb{R}^{n_y}$ on the edges of a $(N \times N)$ grid, so we have $n_y = 2N(N-1)$, and a linear optimization model computing the shortest path from the upper left node to the lower right node of the grid. The optimization problem is still cast as a linear program and solved with Gurobi.

The data is generated following Tang & Khalil (2022). The cost of an edge of the graph representing the square grid is given by

$$(\theta_i)_j = \left[\frac{1}{3.5^4} \left(\frac{1}{\sqrt{n_x}} (Bx_i)_j + 3 \right)^4 + 1 \right] \cdot \epsilon_{ij}, \tag{12}$$

where $B \in \mathbb{R}^{n_y \times n_x}$ is a random matrix whose each element follows a Bernoulli distribution with parameter 0.5, and $\epsilon_{ij} \sim U(0.5, 1.5)$. The contextual vectors x_i follow a standard multivariate Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{I}_{n_x})$.



(a) Number of iterations for varying contextual dimension

(b) Number of iterations for varying decision dimension

(c) Number of iterations for varying model complexity (depth of feed-forward neural network predictor)

Figure 11. Experiment results with tabular data on the shortest paths on a grid problem, showing the sensitivity of our algorithm to the number of (a) features, (b) decisions, and (c) layers

Sensitivity to contextual dimension. First, we analyze the effect of the number of contextual features on the computation time for each explanation type. For each number of contextual features $n_x \in \{5, 10, 25, 50, 100, 500\}$, we generate data: contextual vectors $(x_i)_{i=1}^{2000}$, associated costs $(\theta_i)_{i=1}^{2000}$ and shortest paths $(y_i)_{i=1}^{2000}$. The grid size if fixed to (5×5) (that is, $N = 5$ and $n_y = 40$). Then, we train a linear regression model with the SPO+ loss (Elmachtoub & Grigas, 2022) for 70 epochs (until convergence) on the 1000 first data (train set), with Adam (Kingma & Ba, 2015) and a 3×10^{-4} learning rate.

In each setting, we perform 100 explanations of each type, on 100 random pairs (x_i, y_j), belonging to the 1000 last generated data (test set). x_i plays the role of x_0 (initial context), and y_j plays the role of y^{alt} (alternative solution). We check that $y_i \neq y_j$ for relative and absolute explanations to ensure that the explanation problem is not trivial (ε-explanations do not require an y^{alt} anyway). ε-explanations are performed with $\varepsilon = 1$. We use a step size $\gamma = 0.1$, maximum number of iterations $K = 6000$, maximum number of non-improving iterations $c_{\max} = 10$, and update tolerance $u = 0.9$. The results are displayed in Figure 11a, and show that explanations can be obtained efficiently even for large numbers of features.

Sensitivity to decision dimension. We now analyze the effect of the size of the costs grid (and thus the number of decision features n_y) on the computational time for each explanation type. The data generation and training processes are unchanged w.r.t. the above experiment. We choose a fixed number of contextual features $n_x = 10$, and experiment for $N \in \{2, 3, 4, 5, 7, 10\}$ (corresponding to $n_y \in \{4, 12, 24, 40, 84, 180\}$). The results are shown in Figure 11b. The figure shows that obtaining relative and ε-explanations scales well in the number of decisions, contrary to absolute explanations. This is caused by the fact that the grid size is the source of the combinatorial explosion of the number of solutions.

Sensitivity to prediction model's complexity. Then, we analyze the effect of the prediction model's complexity, by using a fully connected feed-forward ReLU neural network as φ_L, and observing the evolution of computation times for different numbers of layers (denoted L). The number of contextual features is fixed to $n_x = 10$, and the grid size to (5×5) (that is, $N = 5$ and $n_y = 40$). The first layer has input dimension n_x and output dimension n_x (n_y if $L = 1$, which boils down to the linear regression model), and all other layers have input and output dimension equal to n_x except the output layer, which has output dimension n_y. The train/test dataset is generated only once, with 10000 data points in the first and 1000 in the latter. For $L \in \{1, 2, 3, 4, 5\}$, we train φ_L, whose architecture is described above, until convergence (before overfitting) on the train set with Adam and a 3×10^{-4} learning rate. The results are shown in Figure 11c, and the number of iterations needed to obtain explanations is almost independent of the model complexity for relative and ε explanations.

Varying ε for ε-explanations. We finally analyze the effect of ε on ε-explanation tasks by observing the distance of the obtained explanation to the initial context $||x^{\text{alt}} - x_0||_2^2$. We use a single dataset (train and test), generated with $n_x = 10$ contextual features, a (5×5) grid (that is, $N = 5$ and $n_y = 40$), and the same generating routine as before. The linear regression model is trained with the same process as described above. For $\varepsilon \in \{0.2, 0.5, 0.7, 1, 1.5, 2, 3\}$, we perform 100 ε-explanations on 100 random initial contexts x_0, taken from the test dataset. The results are displayed in Figure 12 and show, as expected, that ε-explanations move away from the initial context as ε increases.

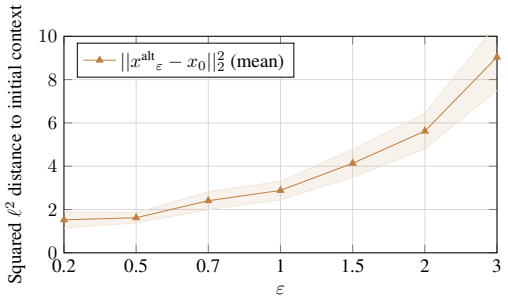


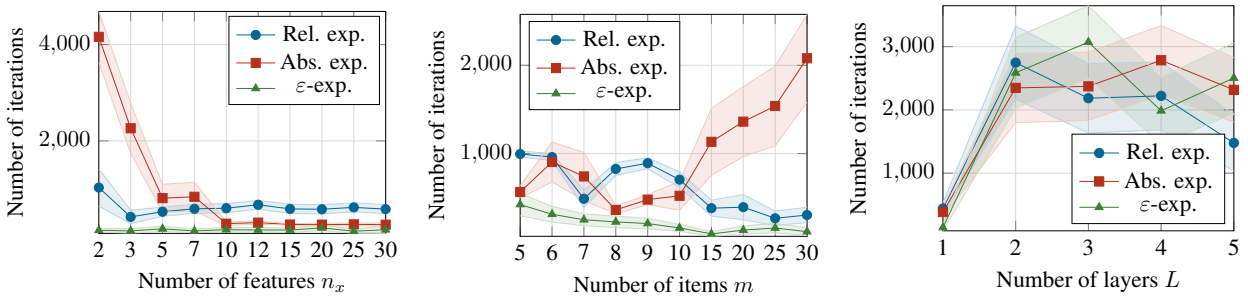
Figure 12. Proximity of ε-explanation $x^{\text{alt}}{}_\varepsilon$ to initial context x_0 for varying ε, measured with squared ℓ^2.

B.2.2. Contextual Multi-Dimensional Knapsack

We evaluate our approach on the contextual multi-dimensional knapsack problem presented in Tang & Khalil (2022), in which the decision-maker maximizes the expected rewards of items to select subject to weight constraints. The relationship between the features and rewards is also given by Equation (12). The structured learning pipeline is identical to the one in the previous experiment. The optimization model is formulated as a binary model, in which each variable indicates whether an item is selected or not, cast as an integer linear program and solved using Gurobi.

Sensitivity to contextual dimension. First, we analyze the effect of the number of contextual features on the computation time for each explanation type on the knapsack problem. For each number of contextual features $n_x \in \{2, 3, 5, 7, 10, 12, 15, 20, 25, 30\}$, we generate data: contextual vectors $(x_i)_{i=1}^{6000}$, associated rewards $(\theta_i)_{i=1}^{6000}$ and optimal solutions $(y_i)_{i=1}^{6000}$. The number of items is fixed to $m = 16$. Then, we train a linear regression model with the SPO+ loss (Elmachtoub & Grigas, 2022) for 30 epochs (until convergence) on the 5000 first data (train set), with Adam (Kingma & Ba, 2015) and a 1×10^{-3} learning rate.

In each setting, we perform 100 explanations of each type, on 100 random pairs (x_i, y_j), belonging to the 1000 last generated data (test set). x_i plays the role of x_0 (initial context), and y_j plays the role of y^{alt} (alternative solution). We check that $y_i \neq y_j$ for relative and absolute explanations to ensure that the explanation problem is not trivial (ε-explanations



(a) Number of iterations for varying contextual dimension

(b) Number of iterations for varying number of items

(c) Number of iterations for varying model complexity (depth of feed-forward neural network predictor)

Figure 13. Experiment results with tabular data on the contextual multi-dimensional knapsack problem, showing the sensitivity of our algorithm to the number of (a) features, (b) items, and (c) layers

do not require an y^{alt} anyway). ε-explanations are performed with $\varepsilon = 0.1$. We use a step size $\gamma = 0.01$, maximum number of iterations $K = 6000$, maximum number of non-improving iterations $c_{\max} = 10$, and corresponding update tolerance $u = 0.9$. The results are displayed in Figure 13a. They show once again that explanations can be obtained efficiently even for a larger number of features. The absolute explanations, however, require more iterations on average when the number of features is very low: this is caused by the fact that the prediction model being linear, the set of rewards it can span is limited by its rank (and thus by its input dimension), leading to infeasible absolute explanation tasks which increase the average computation time by taking $K = 6000$ iterations (the maximum) to stop.

Sensitivity to the number of items. We now analyze the effect of the number of items m on the computational time for each explanation type. The data generation and training processes are unchanged w.r.t. the above experiment. We choose a fixed number of contextual features $n_x = 5$ and experiment for $m \in \{5, 6, 7, 8, 9, 10, 15, 20, 25, 30\}$. The results are shown in Figure 13b. The figure highlights that obtaining relative and ε-explanations scales well in the number of items, contrary to absolute explanations This is caused by the fact that the number of items greatly improves the combinatorial difficulty of the problem.

Sensitivity to the prediction model's complexity. Then, we analyze the effect of the prediction model's complexity by using a fully connected feed-forward ReLU neural network as φ_L, and observing the evolution of computation times for different numbers of layers (denoted L). The architecture is similar to the one used in the shortest paths on a grid experiment. The train/test dataset is generated only once, with 5000 data points in the first and 1000 in the latter. For $L \in \{1, 2, 3, 4, 5\}$, we train φ_L, whose architecture is described above, until convergence (before overfitting) on the train set with Adam and a 1×10^{-3} learning rate.

The results are shown in Figure 13c, which shows that the number of iterations needed to obtain explanations for the knapsack problem is almost independent of the model complexity when the number of layers exceeds one. When only one layer is used (the predictor then boils down to a linear regression), the average number of iterations required is significantly lower for all types of explanations.

Varying ε for ε-explanations. We finally analyze the effect of ε on ε-explanation tasks for the knapsack problem by observing the distance of the obtained explanation to the initial context $||x^{\text{alt}} - x_0||_2^2$. We use a single dataset (train and test) generated with $n_x = 5$ contextual features, $m = 16$ items, and the same generating routine as before. The linear regression model is trained using the same process as described above. For $\varepsilon \in \{0.2, 0.5, 0.7, 1, 1.5, 2, 3\}$, we perform 100 ε-explanations on 100 random initial contexts x_0, taken from the test dataset. The results are displayed in Figure 14 and show, as expected and as for the shortest paths on a grid experiment, that ε-explanations move away from the initial context as ε increases.

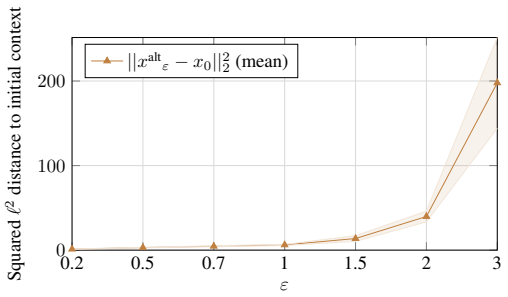


Figure 14. Proximity of ε-explanation $x^{\text{alt}}{}_\varepsilon$ to initial context x_0 for varying ε, measured with squared ℓ^2.

C. Examples of Explanations for the Shortest Paths on Warcraft Maps Pipeline

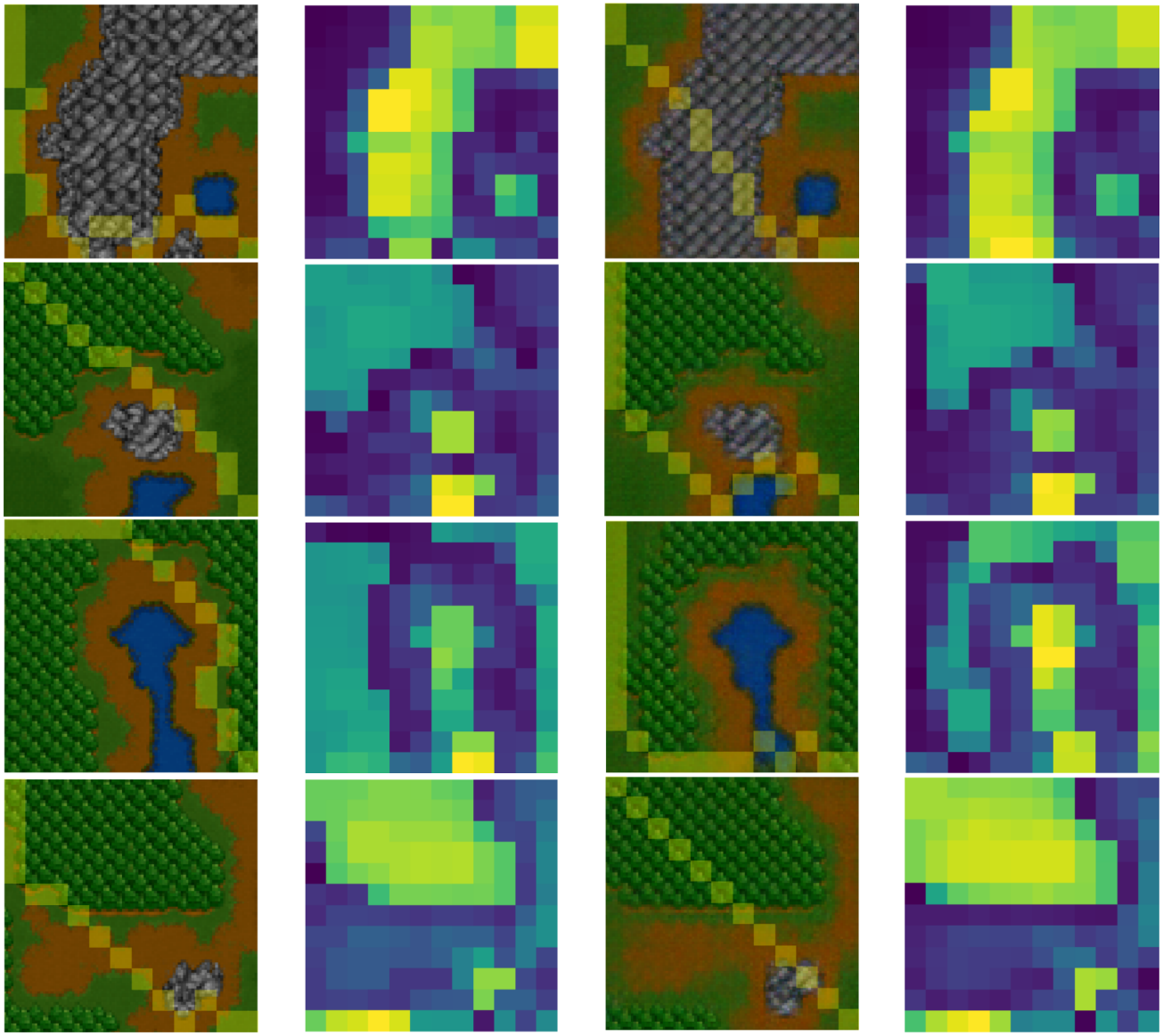
We display several examples of counterfactual explanations for the shortest paths on Warcraft maps structured pipeline. All of them are obtained with a VAE trained in a cost-aware fashion with $\alpha = 2$ and latent hypersphere regularization with $\beta = 10$. We show examples of relative and absolute explanations in Figure 15 and Figure 16 respectively, and show examples of ε-explanations for varying ε in Figure 17.

Consider, for instance, the first row of Figure 15. The initial map on the left shows that the shortest path goes along the gray mountain until the bottom of the map. We ask what a close Warcraft map would be, such that a path crossing the mountain diagonally (i.e., the given alternative solution y^{alt}) is shorter than the initial one, y_0. The computed relative explanation, shown in the 3rd column together with its alternative path y^{alt}, extends the mountain to cover the bottom of the map. Notice that the rest of the image is not modified: this small change is sufficient to render the alternative path less costly than the initial one.

Similarly, the mountain is extended to the top of the image in the second row of Figure 16. Again, the rest of the map is unchanged, and this small modification is enough to make the alternative diagonal path optimal (and not just shorter than the initial one, since we now ask for an absolute explanation). The explanations are not only adding mountain areas to increase travel times. They adequately capture the dataset's structure and its impact on the decision-making process. In the second row of Figure 15, for instance, the explanation is obtained by removing the forest on the left edge of the map.

Counterfactual explanations provide a sensitivity analysis of the pipeline that is both local (i.e., tailored to a specific input) and decision-focused. They allow the user to contrast two decisions in the feature space and to identify what specific features drive the decision-making process.

The novel type of explanation introduced in our paper (ε-explanations) is especially suitable for identifying features whose small variations will most impact the decision. Consider the example given in Figure 17. Figures (c) to (h) show that the central region of the map is the most sensitive: small changes in the forest blocks make the current decision sub-optimal by a factor of 30%, 50%, and up to 70%. In practice, this could be used as a mechanism to tell decision-makers what features should be given special attention, for instance, to verify that their values are correctly measured.



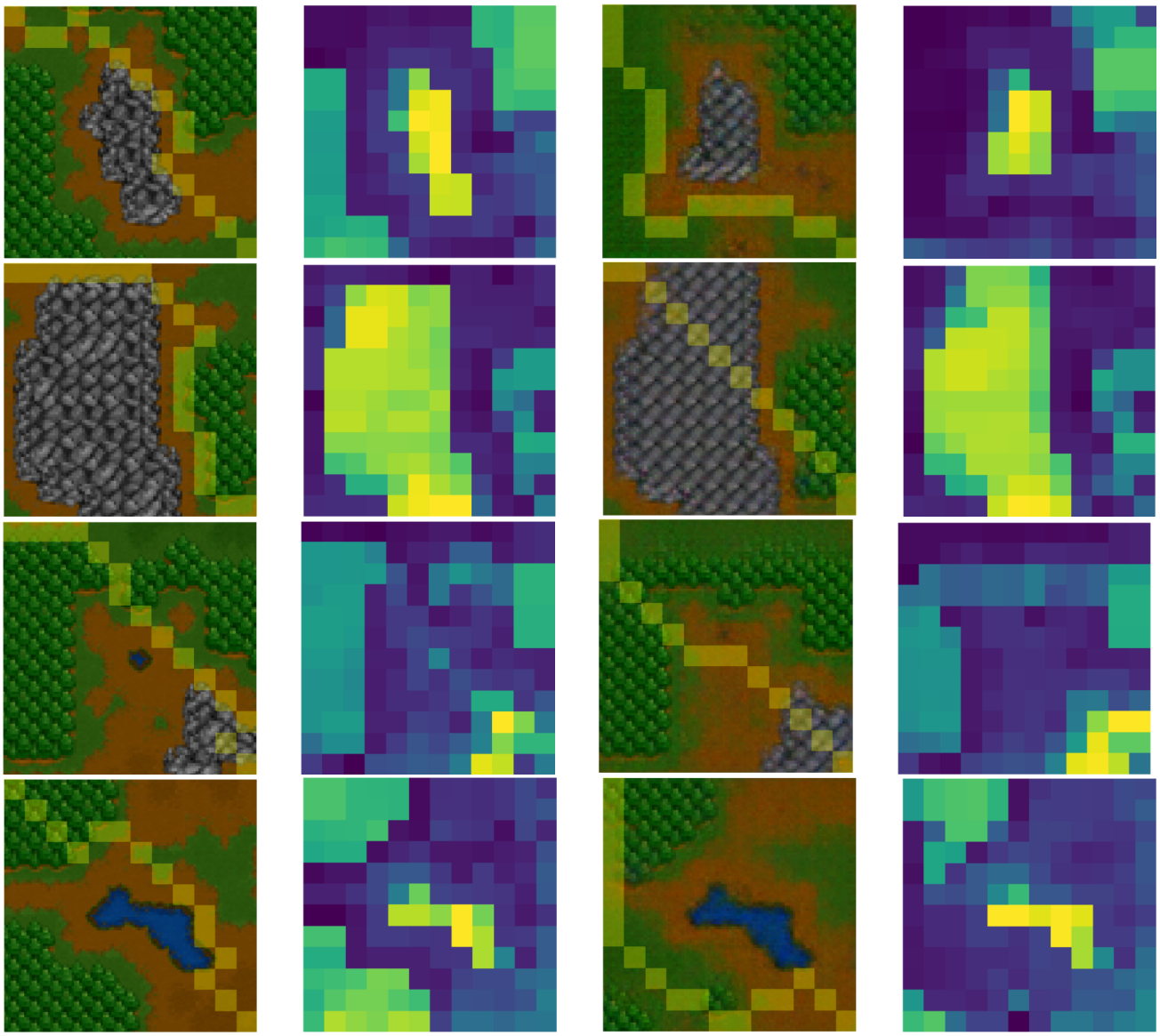
(a) Initial map x_0 and associated shortest path y_0

(b) Initial predicted costs $\varphi(x_0)$

(c) Counterfactual map x^{alt} and alternative path y^{alt}

(d) New predicted costs $\varphi(x^{\text{alt}})$

Figure 15. Example of relative explanations: explanations such that the given alternative path y^{alt} is shorter than the initial shortest one y_0 on the counterfactual map x^{alt}.



(a) Initial map x_0 and associated shortest path y_0

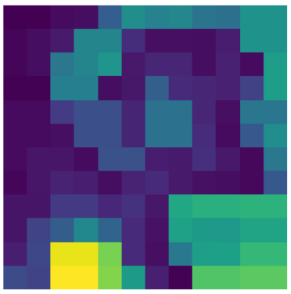
(b) Initial predicted costs $\varphi(x_0)$

(c) Counterfactual map x^{alt} and alternative path y^{alt}

(d) New predicted costs $\varphi(x^{\text{alt}})$

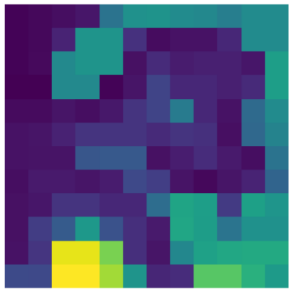
Figure 16. Example of absolute explanations: explanations such that the given alternative path y^{alt} is optimal on the counterfactual map x^{alt}.

(a) Initial map x_0 and associated shortest path y_0



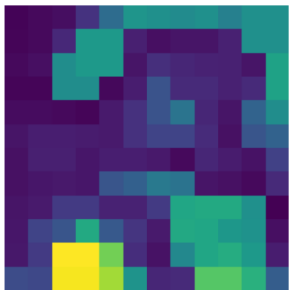
(b) Initial predicted costs $\varphi(x_0)$

(c) ε-explanation with ε=0.3 and associated shortest path

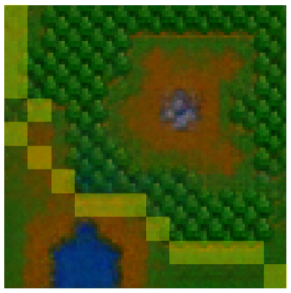


(d) Corresponding predicted costs ($\varepsilon = 0.3$)

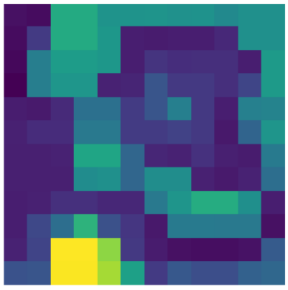
(e) ε-explanation with ε=0.5 and associated shortest path



(f) Corresponding predicted costs ($\varepsilon = 0.5$)

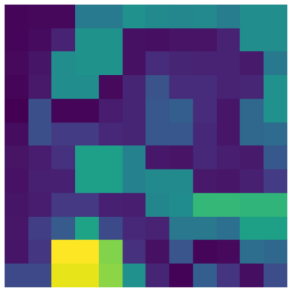


(g) ε-explanation with ε=0.7 and associated shortest path



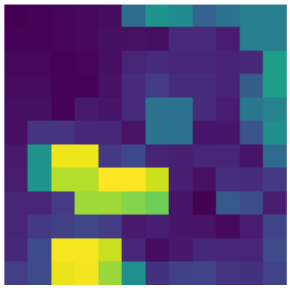
(h) Corresponding predicted costs ($\varepsilon = 0.7$)

(i) ε-explanation with ε=1 and associated shortest path



(j) Corresponding predicted costs ($\varepsilon = 1$)

(k) ε-explanation with ε=2 and associated shortest path



(l) Corresponding predicted costs ($\varepsilon = 2$)

Figure 17. Example of ε-explanations with varying ε: explanations such that the initial shortest path y_0 has relative regret of at least ε on the counterfactual map x^{alt}.

D. Supplementary Material: Algorithms

The latent-space and feature-space formulations of `CF-OPT` are given in Algorithm 1 and Algorithm 2 respectively.

Algorithm 1 Counterfactual Explanations through First-Order Optimization - Latent Space Formulation

Initialize: $z^{(1)} = e_\phi(x_0)$, $\lambda^{(1)} = 0$, $c = 0$, $x_{\text{best}} = \text{None}$, $l_{\text{best}} = \infty$
for $k = 1$ to K **do**
  $x^{(k)} \leftarrow d_\psi\left(z^{(k)}\right)$
  if $c = c_{\max}$ **then**
    Return: x_{best}
  end if
  `/*** Check if improvement ***/`
  if $x^{(k)}$ satisfies the explanation criterion **then**
    if $\ell(x_0, x^{(k)}) + \Omega(z^{(k)}) < u \cdot l_{\text{best}}$ **then**
      Store best solution: $x_{\text{best}} \leftarrow x^{(k)}$
      Store best loss: $l_{\text{best}} \leftarrow \ell(x_0, x^{(k)}) + \Omega(z^{(k)})$
      Reset counter: $c \leftarrow 0$
    else
      Increase counter: $c \leftarrow c + 1$
    end if
  end if
  `/*** Update variables ***/`
  Update latent code: $z^{(k+1)} \leftarrow z^{(k)} - \gamma \nabla_z \mathcal{E}(z^{(k)}, \lambda^{(k)})$
  Update Lagrange multiplier:
  $\lambda^{(k+1)} \leftarrow \lambda^{(k)} + \gamma \nabla_\lambda \mathcal{E}(z^{(k)}, \lambda^{(k)})$
end for
Return: x_{best}

Algorithm 2 Counterfactual Explanations through First-Order Optimization - Feature Space Formulation

Initialize: $x^{(1)} = x_0$, $\lambda^{(1)} = 0$, $c = 0$, $x_{\text{best}} = \text{None}$
for $k = 1$ to K **do**
  if $c = c_{\max}$ **then**
    Return: x_{best}
  end if
  `/*** Check if improvement ***/`
  if $x^{(k)}$ satisfies the explanation criterion **then**
    if $\ell(x_0, x^{(k)}) < u \cdot l_{\text{best}}$ **then**
      Store best solution: $x_{\text{best}} \leftarrow x^{(k)}$
      Store best loss: $l_{\text{best}} \leftarrow \ell(x_0, x^{(k)})$
      Reset counter: $c \leftarrow 0$
    else
      Increase counter: $c \leftarrow c + 1$
    end if
  end if
  `/*** Update variables ***/`
  Update covariate: $x^{(k+1)} \leftarrow x^{(k)} - \gamma \nabla_x \mathcal{E}(x^{(k)}, \lambda^{(k)})$
  Update Lagrange multiplier:
  $\lambda^{(k+1)} \leftarrow \lambda^{(k)} + \gamma \nabla_\lambda \mathcal{E}(x^{(k)}, \lambda^{(k)})$
end for
Return: x_{best}