Ensemble of Counterfactual Explainers

Riccardo Guidotti[1]✉ and Salvatore Ruggieri[2]

University of Pisa, Italy, {name.surname}@unipi.it

Abstract. In eXplainable Artificial Intelligence (XAI), several counterfactual explainers have been proposed, each focusing on some desirable properties of counterfactual instances: minimality, actionability, stability, diversity, plausibility, discriminative power. We propose an ensemble of counterfactual explainers that, as for ensemble of classifiers, boosts weak explainers that provide only a subset of such properties to a powerful method covering all of them. The ensemble runs weak explainers on a sample of instances and of features, and it combines their results by exploiting a diversity-driven selection function. The method is model-agnostic and, through a wrapping based on autoencoders, it is also data-agnostic. A qualitative and quantitative experimentation is presented to validate the approach and to compare with state-of-the-art methods.

Keywords: Counterfactual Explanations · Interpretable Machine Learning · Explainable Artificial Intelligence · Ensemble of Explainers

1 Introduction

Counterfactual explanations are at the highest level of the Judea Pearl's interpretability scale, as they answer why a decision has been made by highlighting what changes in the input would lead to a different outcome. Thinking in counterfactual terms requires imagining a reality that contradicts the observed facts, hence the name "counterfactuals" [15]. Counterfactuals help problem solving by allowing humans to reason on cause-effect relations and what-if scenarios. In eXplainable AI, several counterfactual explainers have been proposed, each focusing on desirable properties of counterfactual instances [19]: availability, validity, minimality, actionability, plausibility, diversity, stability, discriminative power.

Consider an instance x for which a black box decision $b(x)$ has to be explained. It should be possible to find various counterfactual instances c (*availability*) which are *valid* (change the decision outcome, i.e., $b(c) \neq b(x)$), *minimal* (the number of features changed in c w.r.t. x should be as small as possible), *actionable* (the feature values in c that differ from x should be controllable) and *plausible* (the feature values in c should be coherent with the reference population). The counterfactuals found should be similar to x (*proximity*), but also different among each other (*diversity*). Also, they should exhibit a *discriminative power* to characterize the black box decision boundary in the feature space close to x. Counterfactual explanation methods should return similar counterfactuals for similar instances to explain (*stability*). Finally, they must be fast enough (*efficiency*) to allow for interactive usage.

In the literature, these desiderata for counterfactuals are typically modeled through an optimization problem [16], which, on the negative side, favors only a subset of the properties above. We propose here an *ensemble of counterfactual explainers* (ECE) that, as in the case of ensemble of classifiers, boosts weak explainers to a powerful method covering all of them. The ensemble runs *base counterfactual explainers* (BCE) on a sample of instances and of features, and it combines their results by exploiting a diversity-driven selection function. The method is model-agnostic and, through a wrapping based on encoder/decoder functions, it is also data-agnostic. We will be able to reason uniformly on counterfactuals for tabular, images, and time series data types. An extensive experimentation is presented to validate the approach. We compare with state-of-the-art explanation methods on several metrics from the literature.

2 Related Work

Research on eXplainable AI (XAI) has flourished over the last few years [5, 14]. Explanation methods can be categorized as: *(i) intrinsic* vs *post-hoc*, depending on whether the AI model is directly interpretable, or if the explanation is computed for a given black box model; *(ii) model-specific* vs *model-agnostic*, depending on whether the approach requires access to the internals of the black box model; *(iii) local* or *global*, depending on whether the explanation regards a specific instance, or the overall logic of the black box. Furthermore, explanation methods can be categorized w.r.t. the type of explanation they return (factual or counterfactual) and w.r.t. the type of data they work with (tabular data, images, text, time series, etc.). Here, we restrict to local and post-hoc methods returning counterfactual explanations, which is the focus of our proposal.

A recent survey of counterfactual explainers is [19]. Most of the systems are data-specific and generate synthetic (*exogenous*) counterfactuals. Some approaches search *endogenous* counterfactuals in a given dataset [11] of instances belonging to the reference population. Exogenous counterfactuals may instead break known relations between features, producing unrealistic instances. Early approaches generated exogenous counterfactuals by solving an optimization problem [16]. In our proposal, we do not rely on this family of methods as they are typically computationally expensive. Another family of approaches are closer to instance-based classification, and rely on a distance function among instances [11, 12]. E.g., [12] grows a sphere around the instance to explain stopping at the decision boundary of the black box. They are simple but effective, and the idea will be at the core of our base explainers. Regarding distance between tabular instances with both continuous and categorical features, we adhere to common approach of using mixed-distance functions. Some approaches deal with high dimensionality of data through autoencoders [3], which map instances into a smaller latent feature space. Search for counterfactuals is performed in the latent space, and then instances are decoded back to the original space. We rely on this idea to achieve a data-agnostic approach.

3 Problem Setting

A *classifier* b is a function mapping an instance x from a reference population in a feature space to a nominal value y also called class value or decision, i.e., $b(x) = y$. We use $b(X) = Y$ as a shorthand for $\{b(x)|x \in X\} = Y$. The classifier b is a *black box classifier* (in short, a black box) when its internals are either unknown to the observer or they are known but uninterpretable by humans. Examples include neural networks, SVMs, ensemble classifiers [5].

A *counterfactual* is an instance c for which the decision of the black box differs from the one of x, i.e., such that $b(c) \neq b(x)$. A counterfactual is *actionable* if it belongs to the reference population. Since one may not have a complete specification of the reference population, a relaxed definition of actionability is to require the counterfactual to satisfy given constraints on its feature values. We restrict to simple constraints $a_A(c, x)$ that hold iff c and x have the same values over for a set A of *actionable features*. Non-actionable features (such as age, gender, race) cannot be changed when searching for a counterfactual.

A *k-counterfactual explainer* is a function f_k returning a set $C = \{c_1, \ldots, c_h\}$ of $h \leq k$ actionable counterfactuals for a given instance of interest x, a black box b, a set X of known instances from the reference population, and a set A of actionable features, i.e., $f_k(x, b, X, A) = C$. For endogenous approaches, $C \subseteq X$. For machine learning developers adopting explainability methods for debugging black boxes, the natural choice of X is the training set. A counterfactual explainer is model-agnostic (resp., data-agnostic) if the definition of f_k does not depend on the internals of b (resp., on the data type of x). We consider the following data types: tabular data, time series and images. For *tabular data*, an instance $x = \{(a_1, v_1), \ldots, (a_m, v_m)\}$ is a tuple of m attribute-value pairs (a_i, v_i), where a_i is a feature (or attribute) and v_i is a value from the domain of a_i. For example, $x = \{(age, 22), (sex, male), (income, 800)\}$. The domain of a feature can be continuous (*age*, *income*), or categorical (*sex*). For (univariate) *time series*, an instance $x = \langle v_1, \ldots, v_m \rangle$ is an ordered sequence of continuous values (e.g., the body temperature registered at hourly rate). For *images*, an instance x is a matrix $x \in \mathbb{R}^{m \times m}$ whose values represent the intensity of the pixels.

Problem Statement. We consider the problem of designing a k-counterfactual explainer satisfying a broad range of properties: availability, validity, actionability, plausibility, similarity, diversity, discriminative power, stability, efficiency.

4 Ensemble of Explainers

Our proposal to the stated problem consists of an ensemble of base explainers named ECE (Ensamble of Counterfactual Explainers). Ensemble classifiers boost the performance of weak learner base classifiers by increasing the predictive power, or by reducing bias or variance. Similarly, we aim at improving base k-counterfactual explainers by combining them into an ensemble of explainers.

The pseudo-code of ECE is shown in Alg. 1. It takes as input an instance to explain x, the black box to explain b, a set of known instances X, the number

Algorithm 1: ECE

Input : x - instance to explain, b - black box, X - known instances, k - nbr of counterfactuals, A - actionable features, E - base explainers
Output: C - k-counterfactual set

1 $C \leftarrow \emptyset$; // init. result set
2 **for** $f_k \in E$ **do** // for each base explainer
3 $X' \leftarrow \mathcal{I}(X)$; // sample instances
4 $A' \leftarrow \mathcal{F}(A)$; // sample features
5 $C \leftarrow C \cup f_k(x, b, X', A')$; // call base explainer
6 $C \leftarrow \mathcal{S}(x, C, k)$; // select top k-counterfactuals
7 **return** C;

of required counterfactuals k, the set of actionable features A, a set of base k-counterfactual explainers E, and it returns (at most) k counterfactuals C. Base explainers are invoked on a sample[1] without replacement X' of instances from X (line 3), and on a random subset A' of the actionable features A (line 4), as in Random Forests. All counterfactuals produced by the base explainers are collected in a set C (line 5), from which k counterfactuals are selected (line 6). Actionability of counterfactuals is guaranteed by the base explainers[2]. Diversity is enforced by randomization (instance and feature sampling) as well as by tailored selection strategies. Stability is a result of combining multiple base explainers, analogously to the smaller variance of ensemble classification w.r.t. the base classifiers. Moreover, if all base explainers are model-agnostic, this also holds for ECE. A few base explainers are introduced next, whilst selection strategies $\mathcal{S}$ are discussed in Sect. 4.2. Finally, Sect. 4.3 shows how is possible to apply ECE on any data type by wrapping it around encoding/decoding functions.

4.1 Base Explainers

In the following, we write $X_{(cond(X,\ldots))}$ to denote the instance $x' \in X$ such that $cond(x', \ldots)$ holds. E.g., $X_{(b(X)\neq b(x))}$ is the set of instances $x' \in X$ such that $b(x') \neq b(x)$. We denote by $X_{[S]}$ the projection of the instances in X over the features in S. E.g., $X_{[A]}$ projects over the actionable features A. All BCE's presented are parametric to a distance function $d()$ over the feature space. We write $d_{[A]}(x_1, x_2)$ to denote the distance between x_1 and x_2 over the space of actionable features A. In the experiments, we adopt: for tabular data, a mixed distance weighting Euclidean distance for continuous features and Jaccard dissimilarity for categorical ones; for images and times series, the Euclidean distance.

Brute Force Explainer (bce-b). A brute force approach is illustrated in Alg. 2. BCE-B considers all subsets of actionable features A with cardinality at most n (the limited powerset $\mathcal{P}_n(A)$ in line 2). Also, for each actionable feature, an equal-width binning into r bins is computed (line 3), and for each bin the center value will be used as representative of the bin. The binning scheme

[1] Details of default settings in `https://github.com/riccotti/ECE`

[2] There is no loss of generality in assuming actionability of results of base explainers. If it does not hold, non-actionable counterfactuals can readily be filtering out.

Algorithm 2: BCE-B

Input : x - instance to explain, b - black box, X - known instances, k - nbr of counterfactuals, A - actionable features, n - max nbr features, r - nbr feature values intervals, d - distance fun.
Output: C - k-counterfactual set

1 $C \leftarrow \emptyset$; // init. result set
2 $\mathcal{A} \leftarrow \mathcal{P}_n(A)$; // subsets of size at most n
3 $\mathcal{R} \leftarrow reprvalues(X_{(b(X)\neq b(x)),[A]}, r)$; // discretize values
4 $Z \leftarrow generate(x, \mathcal{A}, \mathcal{R})$; // generate all variations of x
5 $D \leftarrow d_{[A]}(x, Z)$; // calculate distances
6 $Z \leftarrow sort(Z, D)$; // sort w.r.t distances
7 **for** $c \in Z$ **do** // for each candidate
8 $\quad C' \leftarrow refine(c, x, b)$; // improve similarity
9 $\quad C \leftarrow C \cup C'$; // add to result
10 $\quad$ **if** $|C| \geq k$ **then** // if enough counterfactuals
11 $\quad\quad$ **break**; // stop
12 **return** C;

Algorithm 3: BCE-N

Input : x - instance to explain, b - black box, X - known instances, k - nbr of counterfactuals, A - actionable features, d - distance function
Output: C - k-counterfactual set

1 $C \leftarrow \emptyset$; // init. result set
2 $D \leftarrow d_{[A]}(x, X_{(b(X)\neq b(x))})$; // calculate distances
3 $Z \leftarrow sort(X_{(b(X)\neq b(x))}, D)$; // sort w.r.t distances
4 **for** $c \in Z$ **do** // for each candidate
5 $\quad c_{[\sim A]} \leftarrow x_{[\sim A]}$; // overwrite non-actionable features
6 $\quad$ **if** $b(c) \neq b(x)$ **then** // if c is valid
7 $\quad\quad C \leftarrow C \cup \{c\}$; // add to result
8 $\quad$ **if** $|C| \geq k$ **then** // if enough counterfactuals
9 $\quad\quad$ **break**; // stop
10 **return** C;

considers only the known instances X with black box decision different from x (line 3). The brute force approach consists of generating (line 4) all the possible variations of x with respect to any of the subset in $\mathcal{A}$ by replacing an actionable feature value in x with any representative value of a bin of the feature. The distances between x and any variation in Z is calculated (line 5) and variations are ordered by such distance (line 6). For each such variation c, a *refine* procedure (line 8) implements a bisecting strategy of the features in c which are different from x while maintaining $b(c) \neq b(x)$. The procedure returns either a singleton with a counterfactual or an empty set (in case $b(c) = b(x)$). The aim of *refine* is to improve similarity of the counterfactual with x. The procedure stops when k counterfactuals have been found or there is no further candidate (lines 9–11).

Obviously, the greater are n and r, the larger the search space explored by BCE-B. This implies a larger number of counterfactuals to choose from, but also the higher the computational complexity of the approach. The complexity of BCE-B is indeed $O(\binom{|A|}{n} \cdot n \cdot r)$, and it can be mitigated by setting low values for n and r. We will show in the experimental section that $n = 2$ and $r \in \{5, 10, 20\}$ are sufficient for achieving good counterfactuals. While BCE-B tackle minimization of changes and similarity properties, it does not account for diversity.

Nearest Neighbor Explainer (bce-n). A nearest neighbor approach is illustrated in Alg. 3 as simplified version of CBCE [11]. BCE-N calculates the distances between x and the instances in X with black box decision different from x. The distances are calculated over the space of actionable features (line 2). Candidate counterfactuals c are sorted w.r.t. the similarity with x (line 3). Actionability is ensured by overwriting non-actionable features of c with their value in x (line 5). If c is a valid counterfactual, it is added to the result set (lines 6–7). The procedure stops when k counterfactuals have been found or there is no further candidate (lines 8–9). A weakness of BCE-N is the computational cost of computing distances (line 2), which is $O(|X| \cdot |A|)$. Notice, however, that since ECE performs a sampling of the actionable features (line 4), we have that $|A|$ ranges over reasonable small values. This approach only accounts for similarity and validity, but not for diversity and minimization of changes.

Tree-based Explainer (bce-t). The third proposal, shown in Alg. 3, starts from a (surrogate/shadow [7]) decision tree $\mathcal{T}$ trained on X (line 2) to mime the black box behavior. Leaves in T leading to predictions different from $b(x)$ can be exploited for building counterfactuals. Basically, the splits on the path from the root to one such leaf represent conditions satisfied by counterfactuals. To ensure actionability, only splits involving actionable constraints are considered (line 4). To tackle minimality, the filtered paths are sorted w.r.t. the number of conditions not already satisfied by x (line 5). For each such path, we choose one instance c from X reaching the leaf (line 7) and minimizing distance to x (line 8). Even though the path has been checked for actionable splits, the instance c may still include changes w.r.t. x that are not actionable. For this, we overwrite non-actionable features (line 9). Since not all instances at a leaf have the same class as the one predicted at the leaf, we also have to check for validity (line 10) before including c in the result set. The search over different paths of the decision tree allows for some diversity in the results, even though this cannot be explicitly controlled. On the negative side, the computational complexity requires both a decision tree construction (line 2) and a number of distance minimizations (line 8), yet the latter being computed on subsets of X.

Generative Sphere-based Explainer (bce-s). BCE-S, illustrated in Alg. 5, relies on a generative approach growing a *sphere* of synthetic instances around x [12]. Instance are generated in all directions of the feature space until the decision boundary of the black box b is crossed and the closest counterfactual to x is retrieved (lines 1–12). The sphere radius is initialized to η_0 (line 1), and then it is decreased until the boundary is crossed (lines 4-7). Next, a lower bound radius and an upper bound radius are determined such that the boundary of b crosses the area of the sphere between the lower bound and the upper bound radii (line 9-12). In its original version, the growing spheres algorithm generates instances following a uniform distribution. BCE-S adopts instead the *Gaussian-Matched* generation described in [1], which assume Gaussian distributions of the features. The $GM(x_{[A]}, u, a, b)$ function in Alg. 5 generates u instances in the ring centered at $x_{[A]}$ and between radius a and b. To ensure actionability, non-actionable features of generated instances are set as in x (lines 3, 7, 12). Finally,

Algorithm 4: BCE-T

Input : x - instance to explain, b - black box, X - known instances, k - nbr of counterfactuals, A - actionable features, d - distance function
Output: C - counterfactual explanation

1 $C \leftarrow \emptyset$; // init. result set
2 $\mathcal{T} \leftarrow trainTree(X, b(X))$; // train shadow decision tree
3 $P \leftarrow paths(T, pred.class \neq b(x))$; // find counter. paths
4 $P \leftarrow filter(P, x, A)$; // keep actionable paths
5 $P \leftarrow sort(P, x)$; // sort w.r.t number changes
6 **for** $p \in P$ **do** // for each path
7 $Z \leftarrow \mathcal{T}(X, p)$; // get instances in leaf
8 $c \leftarrow \arg\min_{z \in Z} d_{[A]}(x, z)$; // find the most similar
9 $c_{[\sim A]} \leftarrow x_{[\sim A]}$; // overwrite non-actionable features
10 **if** $b(c) \neq b(x)$ **then** // if c is valid
11 $C \leftarrow C \cup \{c\}$; // add to result
12 **if** $|C| \geq k$ **then** // if enough counterfactuals
13 **break**; // stop
14 **return** C;

BCE-S selects from the instances Z in the final ring the ones which are closest to x (line 15) and are valid (lines 17-18). Notice that this selection is equivalent to call BCE-N where $X = Z$. The sphere-based approach accounts for diversity of counterfactuals by generating instances in a ring crossing the decision boundary of b. The complexity of the approach depends on the distance of the decision boundary from x, which in turn determine the number of iterations needed to compute the final ring. Each iteration requires the generation of u instances. Finally, distances are computed for the instances Z in the final ring.

4.2 Counterfactual Selection

The selection function $\mathcal{S}$ at line 5 of Alg. 1 selects top k-counterfactuals from those returned by base explainers. Differently from classifier ensembles, where majority voting is an effective method, in our context there are many distinct counterfactuals returned by the base classifiers, which makes unlikely to have a counterfactual to be returned more than once. As another drawback, diversity of selected counterfactuals is not accounted for by majority voting. In base explainers, diversity can be tackled at instance generation time, e.g., as in [16]. In our case, we have a different problem, which can be formulated as maximizing an optimization function over subsets of valid counterfactuals C:

$$\arg\max_{S \subseteq C \wedge |S| \leq k} h_x(S) \tag{1}$$

We write x as sub-script to denote that $h_x(S)$ depends also on the instance x. We consider two options for $h_x(S)$. The first one, inspired by [16], is *distance-based*:

$$h_x(S) = hd_x(S) = \frac{1}{|S|^2} \sum_{c_i \in S} \sum_{c_j \in S} d(c_i, c_j) - \frac{1}{|S|} \sum_{c \in S} d(c, x) \tag{2}$$

It aims at maximizing the difference between the average distances among the counterfactuals in S (a measure of diversity) and the average distance from x (a

Algorithm 5: BCE-S

Input : x - instance to explain, b - black box, X - known instances, k - nbr of counterfactuals, A - actionable features, d - distance function, u - instances to generate
Output: C - counterfactual explanation

$\eta \leftarrow \eta_0$; // init. radius
$Z \leftarrow GM(x_{[A]}, u, 0, \eta)$; // generate variations of x
$Z_{[\sim A]} \leftarrow x_{[\sim A]}$; // set non-actionable features
while $\exists\, z \in Z.\ b(z) \neq b(x)$ **do**
  $\eta \leftarrow \eta/2$; // decrease radius
  $Z \leftarrow GM(x_{[A]}, u, 0, \eta)$; // generate variations of x
  $Z_{[\sim A]} \leftarrow x_{[\sim A]}$; // set non-actionable features
$lb \leftarrow \eta$; $ub \leftarrow 2\eta$; // init. bounds
while $\nexists\, z \in Z.\ b(z) \neq b(x)$ **do**
  $lb \leftarrow ub$; $ub \leftarrow ub + \eta$; // update bounds
  $Z \leftarrow GM(x_{[A]}, u, lb, ub)$; // generate variations of x
  $Z_{[\sim A]} \leftarrow x_{[\sim A]}$; // set non-actionable features
$C \leftarrow \emptyset$; // init. result set
$D \leftarrow d_{[A]}(x, Z)$; // calculate distances
$Z \leftarrow sort(Z, D)$; // sort w.r.t distances
for $c \in Z$ **do** // for each candidate
  if $b(c) \neq b(x)$ **then** // if c is valid
    $C \leftarrow C \cup \{c\}$; // add to result
  if $|C| \geq k$ **then** // if enough counterfactuals
    break // stop
return C;

measure of similarity). The second one, is *density-based*:

$$h_x(S) = hk_x(S) = |\bigcup_{c \in S} knn_C(c)| - \lambda \sum_{c \in S} d(c, x) \tag{3}$$

It aims at maximizing the difference between the size of neighborhood instances of the counterfactuals (a measure of diversity) and the total distance from x (a measure of similarity) regularized by a parameter λ. $knn_C(c)$ returns the h most similar counterfactuals to c among those in C with respect to the distance function d. Solving the optimization problem (1) is NP-hard in general. Moreover, hd_x and hk_x are not monotone non-negative sub-modular functions, for which the standard greedy algorithm can provide near-optimal theoretical guarantee. However, hk_x is the difference between a monotone non-negative sub-modular function (size of nearest neighbors), and a modular function (distance from x). For such functions, [4] proposes a simple and efficient variant of the standard greedy algorithm, called Cost Scaled Greedy (CSG), which achieves a good theoretical guarantee. We adopt CSG for both hk_x and hd_x.

4.3 Counterfactuals for Other Data Types

We enable ECE to work on any data type by wrapping it around two functions. An *encoder* $\zeta : \mathbb{D} \rightarrow \mathbb{R}^q$ that maps an instance from its actual domain $\mathbb{D}$ to a latent space, and a *decoder* $\eta : \mathbb{R}^q \rightarrow \mathbb{D}$ that maps an instance of the latent space back to the actual domain. Using such functions, any explainer $f_k(x, b, X, A)$ can be extended to the domain $\mathbb{D}$ by invoking $\eta(f_k(\zeta(x), b', \zeta(X), A'))$ where the

	Dataset	n	m	m_{con}	m_{cat}	m_{act}	m_{1h}	l	RF	NN
tabular	adult	32,561	12	4	8	5	103	2	.85	.84
	compas	7,214	10	7	3	7	17	3	.56	.61
	fico	10,459	23	23	0	22	-	2	.68	.67
	german	1,000	20	7	13	13	61	2	.76	.81
img	mnist	60k	28×28	all	0	all	-	10	-	.99
	fashion	60k	28×28	all	0	all	-	10	-	.97
ts	gunpoint	250	150	all	0	all	-	2	-	.72
	power	1,096	24	all	0	all	-	2	-	.98
	ecg200	200	96	all	0	all	-	2	-	.76

Table 1. Datasets description and black box accuracy. n is the no. of instances. m is the no. of features. m_{con} and m_{cat} are the no. of continuous and categorical features respectively. m_{act} is the no. of actionable features. m_{1h} is the total no. of features after one-hot encoding. Rightmost columns report classification accuracy: NN stands for DNN for tabular data, and for CNN for images and time series.

black box in the latent space is $b'(x) = b(\eta(x))$. The definition of the actionable features in the latent space A' depends on the actual encoder and decoder.

Let us consider the image data type (for time series, the reasoning is analogous). A natural instantiation of the wrapping that achieves dimensionality reduction with a controlled loss of information consists in the usage of *autoencoders* (AE) [10]. An AE is a neural network composed by an encoder and a decoder which are trained simultaneously for learning a representation that reduces the dimensionality while minimizing the reconstruction loss. Similarly to [9], we adopt AEs to lift a method designed for the latent space to work on the actual domain. A drawback of this approach is that we cannot easily map actionable feature in the actual domain to features in the latent space (this is a challenging research topic on its own). For this, we set A' to be the whole set of latent features and hence, we are not able to deal with actionability constraints.

5 Experiments

We provide here an empirical analysis of the proposed counterfactual explanation method. Source Python code, the scripts for reproducing the experiments, and an appendix of this paper with a qualitative comparison with other approaches are available at `https://github.com/riccotti/ECE`. Experiments were performed on Ubuntu 20.04 LTS, 252 GB RAM, 3.30GHz x 36 Intel Core i9.

Experimental Setting. We consider a few datasets widely adopted as benchmarks in the literature (see Table 1). There are four tabular datasets[3], where an instance refers to features of an individual person, and the decisions of the black box targets socially sensitive tasks such as income estimation, loan acceptance, risk of recidivism, etc. For each dataset we have selected the set A of actionable features, as follows. `adult`: age, education, marital status, relationship, race, sex, native country; `compas`: age, sex, race; `fico`: external risk estimate; `german`: age, people under maintenance, credit history, purpose, sex, housing,

[3] `https://archive.ics.uci.edu/ml`, `https://www.kaggle.com/datasets`.

foreign worker. There are also two image datasets[4] (`mnist` and `fashion`), and three time series datasets[5] (`gunpoint`, `power` and `ecg200`). Continuous features have been z-score standardized. Categorical features have been one-hot encoded, resulting into m_{1h} total number of features. When presenting counterfactuals, one-hot encoded features are converted back to categorical values.

For each dataset we trained and explained the following black box classifiers: Random Forest (RF) as implemented by *scikit-learn*, and Deep Neural Networks (DNN) implemented by *keras* for tabular datasets, and Convolutional Neural Networks (CNNs) implemented with *keras* for images and time series. We split tabular datasets into a 70% partition used for the training and 30% used for the test, while image and time series datasets are already released in partitioned files. For each black-box and for each dataset, we performed on the training set a random search with a 5-fold cross-validation for finding the best parameter setting[6]. The classification accuracy on the test set is shown in Table 1 (right).

We compare our proposal against several competitors from the state-of-the-art which offer a library[7] that is updated and easy to use. DICE [16] handles categorical features, actionability, and allows for specifying the number k of counterfactuals to return. However, it is not model-agnostic as it only deals with differentiable models such as DNNs. The *FAT* [17] library implements a brute force (BF) counterfactual approach. It handles categorical data but not the number k of desired counterfactuals nor actionability. The *ALIBI* library implements the counterfactual explainers CEM [3,13], CEGP [18] and WACH [20]. All of them are designed to explain DNNs, do not handle categorical features and return a single counterfactual, but it is possible to enforce actionability by specifying the admissible feature ranges. Finally, CEML [2] is a model-agnostic toolbox for computing counterfactuals based on optimization that does not handle categorical features and returns a single counterfactual. We also re-implemented the case-based counterfactual explainer (CBCE) presented in [11]. For each tool, we use the default settings offered by the library or suggested in the reference paper[6].

For each dataset, we explain 100 instances x from the test set. The set X of known instances in input to the explainers is the training set of the black box b. Due to lack of space, we report aggregated results as averages over the 100 instances, datasets and black boxes.

Evaluation Metrics. We evaluate the performances of counterfactual explainers under various perspectives [16]. The measures reported in the following are stated for a single instance x to be explained, and considering $C = f_k(x, b, X, A)$ the returned k-counterfactual set. The metrics are obtained as the mean value of the measures over all x's to explain.

Size. The number of counterfactuals $|C|$ can be lower than k. We define $size = |C|/k$. The higher the better. Recall that by definition of a k-counterfactual explainer, a $c \in C$ is valid, i.e., $b(c) \neq b(x)$.

[4] `http://yann.lecun.com/exdb/mnist`, `https://www.kaggle.com/datasets`.

[5] `https://www.cs.ucr.edu/~eamonn/time_series_data_2018`.

[6] Details of the parameters can be found in the github repository.

[7] DICE `https://github.com/interpretml/DiCE`, *FAT* `https://fat-forensics.org`, *ALIBI* `https://docs.seldon.io/projects/alibi`, CEML `https://github.com/andreArtelt/ceml`.

Actionability. It accounts for the counterfactuals in C that can be realized: $act = |\{c \in C \mid a_A(c, x)\}|/k$. The higher the better.

Implausibility. It accounts for how close are counterfactuals to the reference population. It is the average distance of $c \in C$ from the closest instance in the known set X. The lower the better.

$$impl = \frac{1}{|C|} \sum_{c \in C} \min_{x \in X} d(c, x)$$

Dissimilarity. It measures the proximity between x and the counterfactuals in C. The lower the better. We measure it in two fashions. The first one, named dis_{dist}, is the average distance between x and the counterfactuals in C where the usage of different distance functions d can return different results. The second one, sim_{count}, quantifies the average number of features changed between a counterfactual c and x. The $\mathbb{1}_{cond}$ operator returns 1 if *cond* is true, and 0 otherwise. Let m be the number of features.

$$dis_{dist} = \frac{1}{|C|} \sum_{c \in C} d(x, c) \qquad dis_{count} = \frac{1}{|C|m} \sum_{c \in C} \sum_{i=1}^{m} \mathbb{1}_{c_i \neq x_i}$$

Diversity. It accounts for a diverse set of counterfactuals, where different actions can be taken to recourse the decision of the black box. The lower the better. We denote by div_{dist} the average distance between the counterfactuals in C, and by div_{count} the average number of different features between the counterfactuals.

$$div_{dist} = \frac{1}{|C|^2} \sum_{c \in C} \sum_{c' \in C} d(c, c') \qquad div_{count} = \frac{1}{|C|^2 m} \sum_{c \in C} \sum_{c' \in C} \sum_{i=1}^{m} \mathbb{1}_{c_i \neq c'_i}$$

Discriminative Power. It measures the ability to distinguish through a naive approach between two different classes only using the counterfactuals in C. In line with [6, 16], we implement it as follows. The sets $X_= \subset X$ and $X_\neq \subset X$ such that $b(X_=) = b(x)$ and $b(X_\neq) \neq b(x)$ are selected such that the instances in $X_=, X_\neq$ are the k closest to x. Then we train a simple 1-Nearest Neighbor (1NN) classifier using $C \cup \{x\}$ as training set, and d as distance function. The choice of 1NN is due to its simplicity and connection to human decision making starting from examples. We classify the instances in $X_= \cup X_\neq$ and we use the accuracy of the 1NN as *discriminative power* (*dipo*).

Instability. It measures to which extent the counterfactuals C are close to the ones obtained for the closest instance to x in X with the same black box decision. The rationale is that similar instances should obtain similar explanations [6, 8]. The lower the better.

$$inst = \frac{1}{1 + d(x, x')} \frac{1}{|C||C'|} \sum_{c \in C} \sum_{c' \in C'} d(c, c')$$

with $x' = argmin_{x_1 \in X \setminus \{x\}, b(x_1) = b(x)} d(x, x_1)$ and $C' = f_k(x', b, X, A)$.

Runtime. It measures the elapsed time required by the explainer to compute the counterfactuals. The lower the better.

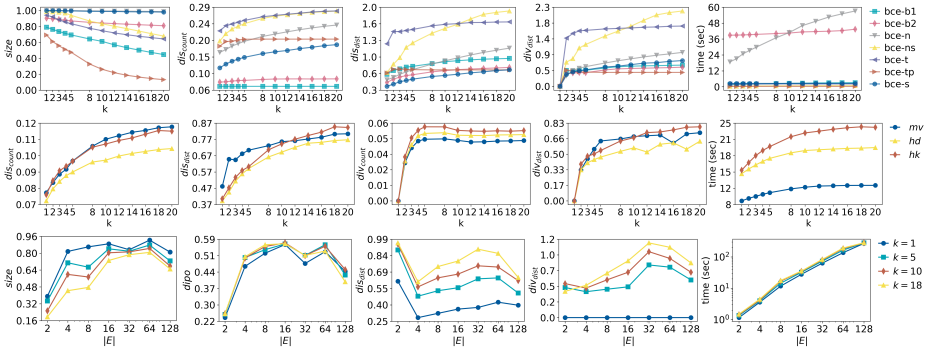

Fig. 1. Aggregated metrics of BCE's by varying k (1^{st} row), of ECE's with different selection functions (2^{nd} row), and of ECE's by varying $|E|$ (3^{rd} row). Y-axis ranges are not uniform across plots. Best view in color.

In line with [7,16,20], in the above evaluation measures we adopt as distance d a mixed distance defined as:

$$d(a,b) = \frac{1}{m_{con}} \sum_{i \in con} \frac{|a_i - b_i|}{MAD_i} + \frac{1}{m_{cat}} \sum_{i \in cat} \mathbb{1}_{a_i \neq b_i}$$

where *con* (resp., *cat*) is the set of continuous (resp., categorical) feature positions. Such a distance is not necessarily the one used by the compared explainers. In particular, it substantially differs from the one used by ECE.

Parameter Tuning. In this section we analyze the performance impact of the components of ECE, in order to fix the default values of its hyper-parameters. The first row in Fig. 1 shows the aggregated performances of some base explainers on tabular datasets when varying the number k of required counterfactuals. The following additional variants are also considered: BCE-B1 and BCE-B2 are BCE-B's with $r = 10$ and max number of features $n = 1$ and $n = 2$, respectively; BCE-NS is BCE-N acting on a random sub-sampling of 100 instances from X as to reduce runtime; and BCE-TP is BCE-T with a decision tree pruned at a maximum height of 4. BCE-S and BCE-N almost always return all the counterfactuals required (first plot), that by design are also valid and actionable. BCE-B1 and BCE-TP on average return less than half of the counterfactuals required. With respect to dis_{count} (second plot) BCE-B1 and BCE-B2 minimize the number of changes w.r.t. x. As one would expect, the metric slowly increases with k. Looking at dis_{dist} (third plot), BCE-S returns counterfactuals similar to x, yet changing more features than BCE-B1 and BCE-B2. Concerning diversity (div_{dist}), BCE-NS and BCE-T are the best performers. Finally, the methods which are clearly less efficient are BCE-B2 and BCE-N. Based on the analysis of those metrics, and on the properties highlighted in Sect. 4, the base explainers used in ECE are set to: BCE-S, BCE-T, and BCE-B1 (from now on, simply BCE-B).

The second row of Fig. 1 shows the aggregate performance of ECE on tabular datasets when varying k and adopting different counterfactual selection functions

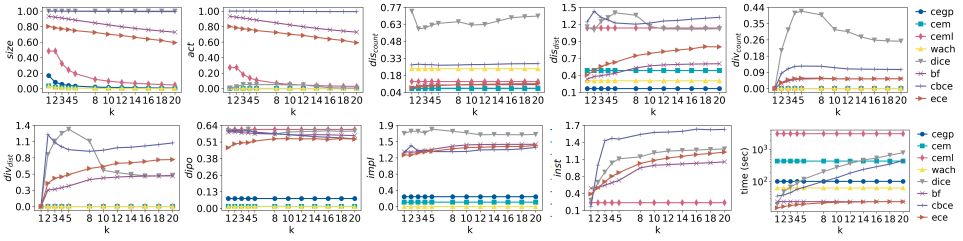

Fig. 2. Aggregate metrics on tabular datasets by varying k.

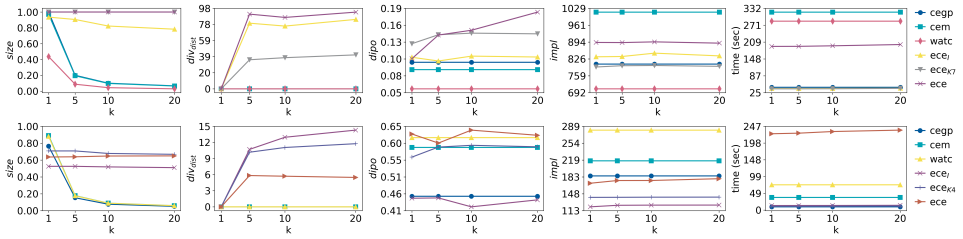

Fig. 3. Aggregate metrics on images (1^{st} row) and time series (2^{nd} row) by varying k.

$\mathcal{S}$: *mv* for majority voting; *hd* for the distance-based (2); *hk* for the density-based (3). The density-based function provides more counterfactuals while requiring slightly higher runtime. The distance-based function has the worst discriminative power. The majority voting function selects counterfactuals randomly in practice, since it is rare a counterfactual is returned more than once. We then fix the density-based function as the default in ECE.

Finally, the third row of Fig. 1 shows the impact of varying the number $|E|$ of BCE's in ECE (the x-axis is in logarithmic scale). An approximately equal number of base explainers are included in E when calling Alg. 1. The first plot suggests to use more than 10 BCE's to find a good fraction of counterfactuals. A similar observation can be made for the discriminatory power *dipo*, while a small number of BCE's is sufficient to minimize the dissimilarity. Diversity has a maximum at about $|E| = 32$. Finally, the runtime grows linearly with $|E|$. In the next experiments, we set $|E| = 10$.

Quantitative Evaluation. Fig. 2 shows the performance of the compared explainers on tabular data when varying k. From the first plot, we notice that only ECE, DICE, CBCE and BF are able to return at least 80% of the required counterfactuals. Most of the other methods only return a single one. From the second plot, we conclude that only ECE, BF and CBCE return a notable fraction of actionable counterfactuals (*act*). DICE and CEML return more than one counterfactual and they allow to specify the actionable features, but they do not actually check their results for validity or actionability. From the plots on dissimilarity (dis_{count} and dis_{dist}) and diversity (div_{count} and div_{dist}), it turns out that CBCE (and also DICE) has good values of diversity, but performs poorly

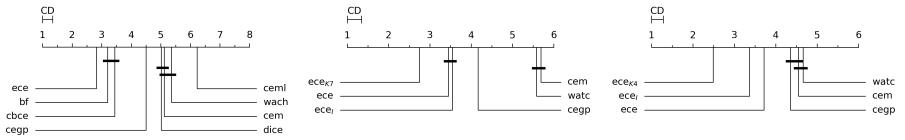

Fig. 4. Critical Difference (CD) diagrams for the post-hoc Nemenyi test at 95% confidence level: tabular (left), images (center), and time series (right) datasets.

w.r.t. dissimilarity. BF wins over ECE w.r.t. the dis_{dist} measure, loses w.r.t. the div_{dist} measure, and is substantially equivalent w.r.t. the other two measures. As for discriminative power *dipo*, ECE performs slightly lower than DICE, CBCE, BF and CEML. Regarding plausibility (*impl*), ECE is the best performer if we exclude methods that return a single counterfactual (i.e., CEM, CEGP and WACH). Indeed, ECE *impl* is constantly smaller that DICE and BF and in line with CBCE, which is the only endogenous methods compared. Intuitively, counterfactuals returned by ECE resemble instances from the reference population. Concerning instability $inst_{x_1,x_2}$, we observe that ECE is slightly worse than BF and slightly better than DICE. CEML is the most stable, and CBCE the most unstable. CEM, CEGP and WACH are not shown in the instability plot because, in many cases, they do not return counterfactuals for two similar instances. Finally, all the explainers, with the exception of BF and ECE, require on average a runtime of more than one minute. We summarize the performances of the approaches by the CD diagram in Fig. 4 (left), which shows the mean rank position of each method over all experimental runs (datasets $\times$ black box $\times$ metrics $\times$ k). Overall, ECE performs better than all competitors, and the difference is statistically significant.

Fig. 3 shows the performance on images (first row) and time series (second row). We consider also ECE with the identity encoder/decoder (named ECE_I), and with the kernel encoder/decoder (ECE_{K7} of size 7×7 and ECE_{K4} of size 4×4). For images, CEM, CEGP and WACH return only a single counterfactual, while ECE provides more alternatives. WACH returns the least implausible counterfactuals, the variants of ECE stand in the middle, while CEM returns less realistic counterfactuals. Regarding running time, CEGP is the most efficient together with ECE_I and ECE_{K4}. The usage of the autoencoder in ECE increases the runtime. CEM and WACH are the slowest approaches. Similar results are observed for time series, with few differences. All the approaches find less counterfactuals and less actionable counterfactuals. ECE returns less diversified sets than ECE_I and ECE_K (see *div* plot). ECE_I and ECE_K return the most plausible sets of counterfactuals and are the two most efficient approaches. The CD diagrams in Fig. 4 (center, right) confirm that ECE and its variants are the best performing methods.

6 Conclusions

We have proposed a model/data agnostic ensemble framework for counterfactual explanation of black box decisions. ECE selects counterfactuals from the results

of base k-counterfactual explainers invoked over sample instances and features. Simple base explainers return valid and actionable counterfactuals. Selection of counterfactual favors diversity. Experimental evaluation has shown that ECE achieves good to best performances w.r.t. several metrics.

Acknowledgment. Work partially supported by the European Community H2020 programme under the funding scheme: H2020-INFRAIA-2019-1: Res. Infr. G.A. 871042 *SoBigData++* and the ERC-2018-ADG G.A. 834756 "XAI: Science and technology for the eXplanation of AI decision making".

References

1. E. Agustsson et al. Optimal transport maps for distribution preserving operations on latent spaces of generative models. In *ICLR*. OpenReview.net, 2019.
2. A. Artelt. Ceml: Counterfactuals for explaining machine learning models - a python toolbox. `https://www.github.com/andreArtelt/ceml`, 2019 - 2021.
3. A. Dhurandhar et al. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In *NIPS*, pages 592–603, 2018.
4. A. Ene et al. Team formation: Striking a balance between coverage and cost. *CoRR*, abs/2002.07782, 2020.
5. R. Guidotti et al. A survey of methods for explaining black box models. *CSUR*, 51(5):1–42, 2018.
6. R. Guidotti et al. Black box explanation by learning image exemplars in the latent feature space. In *ECML-PKDD*, pages 189–205. Springer, 2019.
7. R. Guidotti et al. Factual and counterfactual explanations for black box decision making. *IEEE IS*, 2019.
8. R. Guidotti et al. On the stability of interpretable models. In *IJCNN*, 2019.
9. R. Guidotti et al. Explaining any time series classifier. In *CogMI*, 2020.
10. G. Hinton et al. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.
11. M. T. Keane et al. Good counterfactuals and where to find them. In *ICCBR*, volume 12311 of *LCNS*, pages 163–178. Springer, 2020.
12. T. Laugel et al. Comparison-based inverse classification for interpretability in machine learning. In *IPMU*, pages 100–111. Springer, 2018.
13. R. Luss et al. Generating contrastive explanations with monotonic attribute functions. *arXiv:1905.12698*, 2019.
14. T. Miller. Explanation in artificial intelligence: Insights from the social sciences. *AI*, 267:1 – 38, 2019.
15. C. Molnar. *Interpretable Machine Learning*. Lulu. com, 2020.
16. R. K. Mothilal et al. Explaining machine learning classifiers through diverse counterfactual explanations. In *FAT**, pages 607–617, 2020.
17. K. Sokol et al. Fat forensics: A python toolbox for algorithmic fairness, accountability and transparency. *arXiv:1909.05167*, 2019.
18. A. Van Looveren et al. Interpretable counterfactual explanations guided by prototypes. *arXiv:1907.02584*, 2019.
19. S. Verma et al. Counterfactual explanations for machine learning: A review. *arXiv:2010.10596*, 2020.
20. S. Wachter et al. Counterfactual explanations without opening the black box: Automated decisions and the gpdr. *Harv. JL & Tech.*, 31:841, 2017.

A Reproducibility Settings

For ECE, we set the following parameters unless specified otherwise: the number $|E|$ of explainers is 10, uniformly chosen among BCE-B, BCE-T, and BCE-S; the size of sampled actionable features (line 4 of Alg. 1) is $\sqrt{|A|}$ (rounded down); the size of sampled known instances (line 3) is[8] $|X|/5$. The λ regularization parameter in the density-based selection function (3) is set to $\lambda = 1/(2d_0)$, where $d_0 = min_{c \in C} d(c, x)$. This value is such that at least the counterfactual in C closest to x is worth to be included in the selected set, as the density-based function will be positive. The number h of neighboors considered in the knn_C function in (3) is 5. The distance function d used by ECE and all BCE's is:

$$d(a,b) = \frac{1}{m_{con}} \sqrt{\sum_{i \in con} (a_i - b_i)^2} + \frac{1}{m_{cat}} \sum_{i \in cat} \mathbb{1}_{a_i \neq b_i}$$

where *con* (respectively, *cat*) is the set of continuous (respectively, categorical) feature positions.

Base explainers have the following default parameters. For BCE-B, we set $n = 1$ and $r = 10$, as to favor runtime and similarity of counterfactuals w.r.t. x. For BCE-N, we use the most efficient, yet less accurate, version named BCE-Ns working on a random sample of 100 instances from the known set X. In BCE-T, we do not set any depth/size constraints on the decision tree, as to favor fidelity w.r.t. the black box. For BCE-S, we use $\eta_0 = 4$ as suggested in [12] and $u = 1000$. Since this explainer is heavily dependent on randomness, if a run does not find at least one counterfactual, then the explainer is run again up to a maximum of 10 times. This motivates its high runtime. With respect to the competitors we adopted the default parameter setting or the parameters suggested in their documentation. The following clarification are also worth. DICE is only able to explain differentiable models and therefore was only applied on DNN. CEML is model-agnostic but the tool does not work with `keras`. We could only use it to explain the RF model of `sklearn`. Moreover, CEML is the only explainer that does not exploit any known set X as an input. CEML and DICE allow to specify the desired number of counterfactuals k, the categorical/continuous features, and the set A of actionable features. BF, from the *FAT* library, allows for specifying categorical and continuous features, but not actionable ones nor k. For CEM, CEGP and WACH, from the *ALIBI* library, we could handle actionable features by restricting the values of the non actionable features only to the values in x.

Regarding encoder/decoder functions, we used a VAE with latent space of size 4 for both images and time series. For image datasets, the VAE is composed of a two-layer CNN encoder and decoder with filter sizes $(16, 16)$ and $(32, 32)$ followed by a dense layer with 16 hidden units before the latent layer. For time series datasets, it is composed of a two-layer CNN encoder and decoder with filter sizes 16 and 32 followed by a dense layer with 16 hidden units before the latent layer. The decoder has a symmetric structure with the parameters in the

[8] For experiments where $|E|$ varies, the size is $|X|/min(5, |E|)$.

Algorithm 6: CSG

Input : x - instance to explain,
C - set of counterfactuals,
k - nbr of counterfactuals
Output: S - selected counterfactuals

```
1 S ← ∅                                                  // init. result set
2 for i ∈ [1, k] do                                      // for each required counter.
3 |   s ← arg max_{c∈C} hs'_x(c|S)                        // select best counter.
4 |   if hs'_x(s|S) ≤ 0 then                              // if negative derivative scaled
5 |   |   break                                          // stop
6 |   S ← S ∪ {s}                                        // add to result
7 return S;
```

reverse order. We used *ReLu* as activation function, and *Adam* as optimizer with default `keras` parameters: $learning\_rate = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, minimizing the Kullback–Leibler loss. Autoencoders for images were trained for 1000 epochs, for time series for 100 epochs. In both cases, we adopted an early stopping with patience of 20 epochs.

Alg. 6 shows the Cost Scaled Greedy (CSG) algorithm used by ECE to implement the selection function $\mathcal{S}$ at line 5 of Alg. 1. For a function $h_x(S) = f(S) - c(S)$, the scaled version is simply $hs_x(S) = f(S) - 2c(S)$, and its discrete derivative is:

$$hs'_x(c|S) = hs_x(S \cup \{c\}) - hs_x(S)$$

[4] shows that the solution $S^\star$ to the optimization problem (1) returned by CSG approximates the optimal solution S^{opt} with the following guarantee:

$$f(S^\star) - c(S^\star) \geq 0.5 f(S^{opt}) - c(S^{opt})$$

under the assumptions that f is a monotone non-negative sub-modular function, and c is a modular function. This holds for the density-based function hk_x.

B Qualitative Evaluation

We report here a qualitative comparison of the counterfactuals returned by ECE and the competitor methods, focusing on actionability and on different data types. Fig. 5 shows the counterfactuals returned by the various methods analyzed with $k = 3$ when explaining a DNN or RF on the `german` dataset for an instance x classified as *loan denied*. For each counterfactual, only the feature values different from x are shown. CEM, CEGP and WACH were not able to find any counterfactual (we tried different parameters). CEML only returns one counterfactual, hence not respecting the parameter $k = 3$. DICE returns counterfactuals with many changed features, some of which are not actionable (shown in red). The counterfactuals of CBCE are actionable, but not minimal and quite similar among each other. The ones returned by BF are actionable and minimal but do not foster diversity, as there is one counterfactual repeated twice. Finally, ECE returns minimal, actionable, and diverse results.

Fig. 6 and 7 report the counterfactuals returned when explaining a CNN on `mnist` (Fig. 6), `ecg200` and `power` (Fig. 7) for the instances reported in the top left. In all cases, ECE is the only approach returning more than one counterfactual. For `mnist`, WACH does not return any counterfactual. All the counterfactuals returned by ECE change the black box decision from a '9' to a '7'. As highlighted by counterfactuals c_0 and c_3 of ECE, even a slight modification of the lower part of the ring from white to black causes the black box decision to change from '9' to '7'. CEM shows a counterfactual changing the classification as a '4'. However, this image does not resemble those in the dataset, i.e., it is not plausible. On the other hand, CEGP is in agreement with ECE as returns a '9' with markedly white parts only in the upper pixels of the leg and top of the ring. The ECE considered so far adopts an AE. ECE with the identity encoding (ECE_I) or with kernel encoding (ECE_{K7} is a 7×7 and ECE_{K4} is a 4×4 kernel) return non-plausible counterfactuals, as shown in Fig. 6. Conversely, Fig. 7 shows that using an AE in ECE produces counterfactuals which are valid and plausible but perhaps not very similar to x. The counterfactuals returned by ECE_I and ECE_K appears instead plausible. For the `ecg200`'s image, they show that higher values in the central part after the minimum makes the CNN to change the decision. For `power`, smaller values in the first peak contribute the most to the decision. CEGP counterfactuals are plausible for both cases. The counterfactuals of CEM and WACH in `ecg200` appear distant from x and noisy respectively. Finally, the counterfactual of WACH for `power` is similar to the one of ECE_I.

$x =$ {*duration_in_month = 20, credit_amount = 3271, installment_as_income_perc = 2.97, present_res_since = 2.84, age = 24, credits_this_bank = 1.40, people_under_maintenance = 1, account_check_status = "0 ≤ ··· < 200 DM", credit_history = "existing credits paid back duly till now", purpose = "domestic appliances", savings = "100 ≤ ··· < 500", present_emp_since = "< 1 year", personal_status_sex = "male : married/widowed", other_debtors = "none", property = "if not A121/A122 : car or other, not in attribute 6", other_installment_plans = "none", housing = "rent", job = "skilled employee", telephone = "yes", "foreign_worker = yes"* }

$b(x) =$ *denied loan*

ECE / DNN / RF

c_1 = { *account_check_status = "< 0 DM"* }
c_2 = { *present_emp_since = "1 ≤ ··· < 4 years"* }
c_3 = { *credits_this_bank = 1.98, other_installment_plans = "stores"* }

DICE / DNN

c_1 = { *duration_in_month = 23.91, credit_amount = 4964, installment_as_income_perc = 4.09, present_res_since = 3.39, age = 35.54* }
c_2 = { *duration_in_month = 29.94, credit_amount = 5528, installment_as_income_perc = 4.09, present_res_since = 3.39, age = 35.54, credits_this_bank = 1.84, credit_history = "all credits at this bank paid back duly", purpose = "(vacation - does not exist?)", personal_status_sex = "female : divorced/separated/married", other_debtors = "co-applicant", other_installment_plans = "bank", housing = "for free"* }
c_3 = { *duration_in_month = 29.94, credit_amount = 3835, installment_as_income_perc = 4.09, present_res_since = 3.39, age = 35.54, credits_this_bank = 1.84* }

BF / RF

c_1 = { *telephone = "none"* }
c_2 = { *telephone = "none"* }
c_3 = { *other_debtors = "co-applicant"* }

CEML / RF

c_1 = { *telephone = "none"* }

CBCE / RF

c_1 = { *installment_as_income_perc = 1.85, present_res_since = 1.74, savings = "··· < 100 DM", property = "real estate", telephone = "none",* }
c_2 = { *savings = "··· < 100 DM", present_emp_since = 1 <, telephone = none* }
c_3 = { *account_check_status = "··· ≥ 1000 DM", savings = "··· < 100 DM", present_emp_since = "1 ≤ ··· < 4 years", other_debtors = "co-applicant", property = "real estate", other_installment_plans = "stores", telephone = "none"* }

Fig. 5. Counterfactual explanations for an instance x of the `german` dataset classified as a *denied loan* by DNN or RF. DICE works only on differentiable black boxes. In red, non-actionable changes.

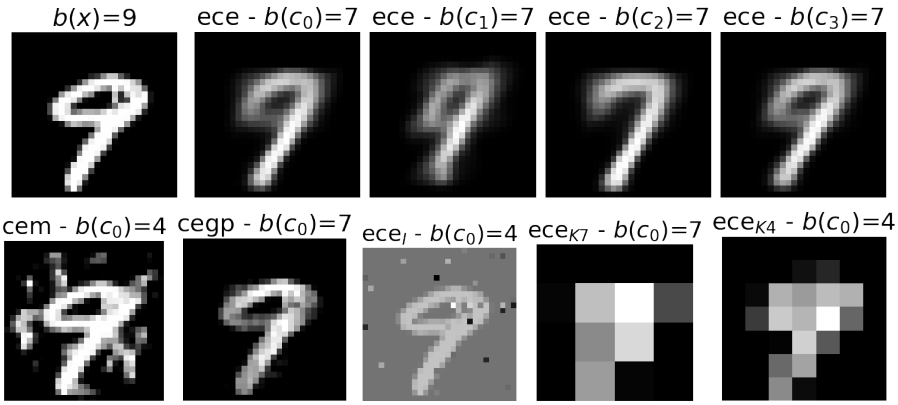

Fig. 6. Counterfactual explanations for an image of the `mnist` dataset classified as 9 (top left) by a CNN.

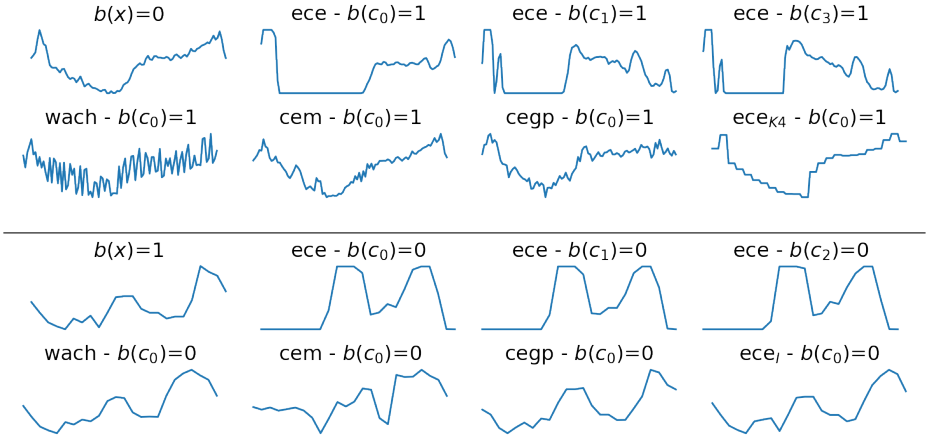

Fig. 7. Counterfactual explanations for two time series of the `ecg200` and `power` datasets classified as 0 and 1 by a CNN (top left).