Data Mining and Knowledge Discovery
https://doi.org/10.1007/s10618-024-01028-9

Discord-based counterfactual explanations for time series classification

Omar Bahri[1] · **Peiyu Li**[1] · **Soukaina Filali Boubrahimi**[1] · **Shah Muhammad Hamdi**[1]

Received: 27 November 2023 / Accepted: 21 April 2024

Abstract

The opacity inherent in machine learning models presents a significant hindrance to their widespread incorporation into decision-making processes. To address this challenge and foster trust among stakeholders while ensuring decision fairness, the data mining community has been actively advancing the explainable artificial intelligence paradigm. This paper contributes to the evolving field by focusing on counterfactual generation for time series classification models, a domain where research is relatively scarce. We develop, a post-hoc, model agnostic counterfactual explanation algorithm that leverages the Matrix Profile to map time series discords to their nearest neighbors in a target sequence and use this mapping to generate new counterfactual instances. To our knowledge, this is the first effort towards the use of time series discords for counterfactual explanations. We evaluate our algorithm on the University of California Riverside and University of East Anglia archives and compare it to three state-of-the-art univariate and multivariate methods.

Keywords EXplainable artificial intelligence · Counterfactual explanations · Time series classification · Time series discords

Responsible editor: Eamonn Keogh.

✉ Omar Bahri
omar.bahri@usu.edu

Peiyu Li
peiyu.li@usu.edu

Soukaina Filali Boubrahimi
soukaina.boubrahimi@usu.edu

Shah Muhammad Hamdi
s.hamdi@usu.edu

[1] Department of Computer Science, Utah State University, 0500 Old Main Hill, Logan, UT 84322-0500, USA

Published online: 07 August 2024

1 Introduction

Counterfactual explanation models have recently emerged as a major player in facilitating explainable artificial intelligence's (XAI) mission to enhance the transparency of machine learning decision systems. In particular, a counterfactual explanation helps stakeholders understand the decision of a complex black-box model and provides actionable insights to alter it by indicating the smallest amount of change required in the input. For example, instead of plainly refusing a mortgage application, a bank could give a justification in the form of a counterfactual explanation, such as: "*if your yearly salary was $25,000 higher, the mortgage would have been approved*". Counterfactual explanations are post hoc, i.e. generated after training the black-box classifier, and are inferred by constructing a synthetic data point that leads to a different model output, called a counterfactual instance.

Despite their popularity in the image and tabular data domains, counterfactual explanation models have had limited success when it comes to time series. Factors such as the lack of cured datasets, the non-intuitive nature of time series data, and its increased complexity are all responsible for this latency (Rojat et al. 2021; Guidotti 2022). The methods available in the literature follow two general approaches. Some of them attempt to generate a counterfactual instance from the original input by introducing features from existing training data. Others introduce new perturbations by solving the problem as an optimization task with different desired properties as parameters. While the first group of approaches generally succeeds in creating realistic counterfactual explanations, the changes they require are usually significantly larger than the optimized perturbations (Guidotti 2022). This trade-off between realistic, plausible and actionable explanations and small perturbations is one of the main challenges in the counterfactual generation process. However, by overly prioritizing its achievement, counterfactual algorithms overlook other fundamental criteria. For example, we show that two of the most popular time series counterfactual generation methods are more likely to fail in finding a valid counterfactual instance given a target class label, i.e. their success rate in terms of "flipping the label" of the original time series instance is lower than 50%. Both methods end up defaulting to an existent instance from the training set, which essentially amounts to a nearest neighbor problem.

We propose DiscoX, a Discord-based counterfactual eXplanation algorithm for time series classification that aims to generate realistic counterfactual instances by using high-level time series shape descriptors to guide the perturbations. DiscoX replaces time series discords in the original time series by their nearest neighbors from the target class. In contrast to other approaches in the literature, DiscoX does not restrict the source of the introduced perturbations to a single training instance, nor does it limit them to their original time steps locations. This allows it to perform better in terms of validity and ensures plausibility since it creates counterfactual instances that are close to the entire space of time series from the target class, not just to a single instance. In addition, by constraining perturbations to the discords' nearest neighbors, it guarantees that the counterfactual instances

remain within a small distance of the original time series. DiscoX is a model agnostic algorithm, meaning that it can be used to explain all types of machine learning models. Moreover, it works for both univariate and multivariate time series.

In addition to validity, there is a crucial need for more user testing of proposed methods (Keane et al. 2021). Delaney et al. (2023) discussed the importance of involving users in the evaluation of counterfactual explanation models. They pointed out the fact that most existing counterfactual approaches have a "class-discrimination" goal, meaning that they primarily aim to find the minimum changes required to alter the class of the original instance. On the other hand, the authors found that people tend to have a "class-distribution" goal where the counterfactual explanations aim to achieve a better understanding of the data domain by highlighting high-level features that are most important for model decisions, regardless of whether the changes are minimal. According to the authors, one way to resolve this "divergence between human and machine counterfactuals" is to shift the focus from the "class-discrimination" by formulating more natural, task-specific goals. Given that the explanation process of DiscoX relies on the use of time series discords as high-level shape descriptors, it represents a step towards "class-distribution" algorithms. Since this paper aims to prove the utility of the method on benchmark datasets, discords are selected based on their MP values. However, by involving users or domain experts in the procedure, specific discords can be selected depending on the explanation goals. We leave this as an avenue for future work.

The rest of this paper is organized as follows. First, we discuss counterfactual generation methods in the literature. Second, we define the counterfactual generation problem and provide important background. Finally, we assess the performance of DiscoX on datasets from the University of California Riverside (UCR) (Dau et al. 2018) and University of East Anglia (UEA) (Hoang et al. 2018) archives and compare it to state-of-the-art algorithms using two different black-box classifiers. The results show that DiscoX achieves better results in terms of validity, sparsity, and proximity without compromising on plausibility. In addition, we show that the perturbations introduced by DiscoX are more meaningful through two different case studies.

2 Related works

Wachter et al. (2018) proposed the first counterfactual explanation model in the context of XAI. It generates explanations through the minimization of a loss function comprising two key components. The first component is a prediction term, which aims to guide the generated counterfactual towards the desired target class label. The second component is a distance term, designed to ensure that the counterfactual instance remains in close proximity to the original instance. Other methods attempt to improve the original loss function (Mothilal et al. 2020; Dhurandhar et al. 2018; Van Looveren and Klaise 2019). For example, Counterfactual Explanations Guided by Prototypes (CEGP) (Van Looveren and Klaise 2019), incorporates a prototype term to the loss function to speed-us the search process and ensure plausibility. DICE

(Mothilal et al. 2020) favors the generation of diverse counterfactual instances by adding a regularization constraint to the loss function. The contrastive explanation method (CEM) (Dhurandhar et al. 2018) enforces plausibility by keeping the counterfactual instances within the target class data manifold using an autoencoder-based term in the loss function.

In the context of time series classification, only a few counterfactual explanation methods are available to this date. A recent survey of XAI methods for time series classification Theissler et al. (2022) identified four approaches in the literature. Among them, Native Guide (NG) by Delaney et al. (2020) achieves good results on univariate data. NG works by introducing perturbations to the original data instance at its most influential time steps, guided by its nearest unlike neighbor (nun) from the target class. Karlsson et al. (2020) generate counterfactual explanations for k-Nearest Neighbors and Random Shapelet Forest Karlsson et al. (2016) as black-box models by either applying a global transformation or a series of local transformations with the goal of introducing the least amount of changes to the original time series. Labaien et al. (2020) showed that CEM gives good results on time series datasets. For multivariate time series, CoMTE (Ates et al. 2021) builds the KD-tree of the target class to identify the nun of the original time series. Then, it employs a heuristic search method involving hill climbing and a subsequent trimming step to identify the smallest necessary set of dimensions to substitute from the nun. If this strategy is unsuccessful in generating a valid counterfactual instance, a greedy search is carried out instead.

More recently, Looveren et al. (2021) proposed a general framework based on conditional generative models that supports different data modalities including time series. Höllig et al. (2022) introduced TSEvo, a method for univariate and multivariate time series based on the Non-Dominated Sorting Genetic Algorithm (Deb et al. 2002) that creates explanations using three different mutations. Filali Boubrahimi and Hamdi (2022) extend the approach by Wachter et al. (2018) through the introduction of a Dynamic Barycenter Averaging (DBA) (Petitjean et al. 2011) loss term and Li et al. (2022a) by guiding the perturbations using a shapelet-based term. MG-CF (Li et al. 2022b) and SETS (Bahri et al. 2022a) are two other methods that base their perturbation process on time series shapelets. Given that both of these methods generate explanations by replacing shapelets in the original time series with subsequences from the training data and since it is possible to leverage the MP to discover shapelets (Zhu et al. 2020), they are most similar to DiscoX. However, none of them uses the MP to mine shapelets. In addition, the method suggested by Zhu et al. (2020) only allows identifying shapelet candidates. It does not guarantee the extraction of subsequences that are maximally representative of a dataset class. The reason is that the MP values are not constrained by the total number of occurrences. Therefore, relying on it will result in subsequences that only occur twice in the dataset, whereas a shapelet should in theory be present in several dataset instances of the same class. Moreover, the MP-based shapelet candidates mining procedure cannot be extended to multivariate time series given that a multivariate MP is not simply equivalent to a matrix of stacked univariate MPs (Yeh et al. 2017a). Finally, Bahri et al. (2022b) proposed TeRCE, a method that mines temporal

rules in multivariate time series and generates explanations by replacing them using data from a nun.

3 Preliminaries

3.1 Problem formalism

Consider a dataset with N multivariate time series instances $\mathcal{D} = \{T_1, T_2, \ldots, T_N\}$, such that each $T_i \in \mathbb{R}^{D \times L}$, where D is the number of dimensions and L is the length of the time series (i.e. number of time steps) is mapped to a class from the mutually exclusive set $\mathcal{C} = \{C_1, C_2, \ldots, C_M\}$. Given a black-box classification model with prediction function $f : \mathbb{R}^{D \times L} \rightarrow \mathcal{C}; T \mapsto C$ trained on the dataset and an instance $T = (T^{(1)}, T^{(2)}, \ldots, T^{(D)}) \in \mathcal{D}$: $T^{(d)} \in \mathbb{R}^L$ with class prediction $f(T) = C_m$, a counterfactual explanation is generated by introducing a perturbation $\Delta = (\Delta^{(1)}, \Delta^{(2)}, \ldots, \Delta^{(D)}) \in \mathbb{R}^{D \times L}$: $\Delta^{(d)} \in \mathbb{R}^L$ to T to achieve $f(T + \Delta) = C_{m'}$ where $C_{m'} \neq C_m$ is the target counterfactual class.

3.2 Counterfactual desired properties

The most important properties that are commonly sought in counterfactual explanations throughout the literature are proximity, sparsity, plausibility, and validity. Achieving a trade-off between all criteria is the primary concern when introducing the perturbation Δ.

- **Proximity:** The counterfactual instance has to be close to the original time series instance. The L_1-, L_2-, and L_{inf}-norms have all been used as proximity measures (Delaney et al. 2020; Mothilal et al. 2020; Van Looveren and Klaise 2019; Filali Boubrahimi and Hamdi 2022; Bahri et al. 2022b). Formally, $||\Delta||_{p \in \{1,2,\infty\}}$ has to be minimal.
- **Sparsity:** The counterfactual instance has to preserve as many features as possible from the original time series instance. Therefore, the number of features modified by Δ have to be minimized (Delaney et al. 2020; Mothilal et al. 2020; Van Looveren and Klaise 2019; Filali Boubrahimi and Hamdi 2022; Bahri et al. 2022b; Karimi et al. 2020). Formally, $||\Delta||_0$ which represents the sum of non-zero elements of Δ has to be minimal.
- **Plausibility:** The counterfactual instance have to be realistic. Thus, they must lie within the data manifold of the training data. The use of autoencoders and outlier detection methods such as the local outlier factor (Breunig et al. 2000; Kanamori et al. 2020), isolation forests (Liu et al. 2008), and one-class support vector machine (Schölkopf et al. 2001) have occurred in the literature (Van Looveren and Klaise 2019; Delaney et al. 2020; Bahri et al. 2022b)
- **Validity:** Since the counterfactual generation processes is not guaranteed to succeed for most methods in the literature, a common evaluation criteria is

to consider the ability to generate a valid counterfactual explanation for the original time series T_o given a target class $C_{m'}$, i.e. the ability to "flip the label" (Mothilal et al. 2020).

3.3 Matrix profile

The Matrix Profile (*MP*) time series data structure, has been introduced recently by Yeh et al. (2017b) and was met with huge success in the time series mining community. By considerably reducing the time complexity required for the similarity search ($\mathcal{O}(L \times \log L)$ using the Mass algorithm (Yeh et al. 2017b)) and the all-pairs similarity search for time series subsequences ($\mathcal{O}(L^2)$ using the STOMP algorithm (Zhu et al. 2018)), the MP has been quickly leveraged to solve a plethora of problems. For example, it has been used for motif discovery (Yeh et al. 2017b), anomaly detection (Lu et al. 2022), and clustering (Gharghabi et al. 2018). However, to the extent of our knowledge, the MP has not been used yet in the context of XAI. DiscoX leverages the MP to 1) find discords from the original instance in regard to the target class instances and 2) match these discords to their closest subsequences.

The *MP* is classically defined for univariate time series. While there have been efforts to extend the structure to multivariate time series (Yeh et al. 2017b), it is not relevant for the purpose of this work. Therefore, this section assumes $D = 1$. Let T be a univariate time series vector of length L: $T = (t_1, t_2, \ldots, t_L)$ and T' a univariate time series vector of length L': $T' = (t'_1, t'_2, \ldots, t'_{L'})$. If T is the query time series and T' is the reference time series, the aim of the MP is to provide fast access to the nearest neighbor of each subsequence of T of length m defined as $sub^T_{i,m} = (t_i, t_{i+1}, \ldots, t_{i+m})$ from the set of T' subsequences $sub^{T'}_{j,m}$ of the same length m. We summarize the main steps as follows:

1. First, for each query subsequence $sub^t_{i,m}$, the distance profile vector $D_i = (d_{i,0}, d_{i,1}, \ldots, d_{i,L-m+1})$ is computed using a sliding window: for each $sub^t_{i,m}$ in T, the distance $dist(z(sub^t_{i,m}), z(sub^{t'}_{j,m}))$ where z is the z-normalization function and $dist$ is the Euclidean distance is computed and stored in $d_{i,j}$. Using the distance profile D_i, finding the distance of the query $sub^t_{i,m}$ to its nearest neighbor simply amounts to finding the smallest value $d_{i,j}$ in D_i. In which case, the nearest neighbor is the subsequence $(t'_j, t'_{j+1}, \ldots, t'_{j+m})$.

   In case the query time series is the same as the reference one $T = T'$, $sub^t_{i,m}$ will also be a subsequence of T'. Therefore, an exclusion zone is first set up around its position i in D_i to avoid trivial matches, i.e. to avoid returning the query itself or an overlapping neighboring subsequence. Traditionally, the exclusion zone is set to $m/2$ on both sides of i.
2. Using the distance profiles D_i of each $sub_{i,m}$, the matrix profile vector $MP(T, T') = (mp_1, mp_2, \ldots, mp_{L-m+1})$ is created such that $mp_i = min(D_i)$. Therefore, $MP(T, T')$ provides direct access to the distance separating each query subsequence $sub^t_{i,m}$ from its nearest neighbor in T'.

3. Finally, the matrix profile index vector $MPI(T, T') = (mpi_1, mpi_2, \ldots, mpi_{L-m+1})$ is created to keep track of the locations of the nearest neighbors of each subsequence, by recording the index of the minimum distance in each D_i: $mpi_i = \texttt{argmin}(D_i)$.

3.4 Time series discords

A time series discord is defined as the subsequence that has the maximum distance to its nearest neighbor. Therefore, discords in T with regard to reference T' represent shape descriptors that characterize T but do not occur in T'.

3.4.1 Univariate time series discords

The *MP* facilitates the extraction of discords in the same way it does for motifs; they occur at its peaks. Given T and T', the most important discord can be simply extracted as the subsequence with the highest value in $MP(T, T')$, i.e. $sub^T_{i,m}$: $i = \texttt{argmax}(MP(T, T'))$ and its nearest neighbor as $sub^{T'}_j$: $j = MPI(T, T')_i$.

3.4.2 Multivariate time series discords

In this section, we describe the process of finding discords in a multivariate time series setting. Given query T and reference T' with $T, T' \in \mathbb{R}^{D \times L}$, the most intuitive way is to compute $MP^d(T, T')$ at each dimension $d \in D$, aggregate the results by summing up the values at each time step to form an aggregate vector $\widehat{MP}(T, T') = \sum_{k=d}^{D} MP(T^{(d)}, T'^{(d)})$, and retrieve the most important discord as the highest value. However, this approach is bound to fail in detecting meaningful discords in real-life datasets. The reason is that in the presence of a high number of dimensions, discords tend to appear in a subset of D only, in which case the rest of the dimensions that do not contribute to the discord will curtail the sum value in $\widehat{MP}(T, T')$.

Tafazoli and Keogh (2023) discuss this issue and propose the TSADIS (Time Series Anomaly Detection through Incremental Search) algorithm to overcome it. We provide an outline of TSADIS in the next paragraph (we highly encourage reading the original paper for a more detailed and illustrated walkthrough). The algorithm was developed for anomaly detection using a discord-based approach. To remain relevant with our paper, we substitute the word "*anomaly*" in the terminology of the original paper with "*discord*". In addition, we adapt the algorithm to the outer-join setting of the current work (i.e. $T \neq T'$), whereas the original paper considers self-joins only (i.e. $T = T'$).

0. TSADIS defines a K-dimensional-discord (KDD) as a discord that appears on at least K dimensions. kDD is a natural discord if k corresponds to its number of natural dimensions, i.e. to the maximum number of dimensions that contribute to it.

1. TSADIS starts by computing the all-matrix-profiles structure $MP_s(T, T') = (MP^{(1)}, MP^{(2)}, \dots, MP^{(D)})$: $MP^{(d)} = MP(T^{(d)}, T'^{(d)})$.
2. Then, it constructs the K-dimensional-discord-score (KS) lists, where each kS list contains all possible combinations of discord-score sets (S) of size k. A discord-score set is any combination of discord scores ($MP^{(d)}$) from $MPs(T, T')$. For T and T', the kS lists are formally represented as follows:

$$1S = (\{MP^{(1)}\}, \{MP^{(2)}\}, \dots, \{MP^{(D)}\})$$
$$2S = (\{MP^{(1)}, MP^{(2)}\}, \{MP^{(1)}, MP^{(3)}\}, \{MP^{(2)}, MP^{(3)}\}, \dots, \{MP^{(D-1)}, MP^{(D-1)}\})$$
$$\dots$$
$$DS = (\{MP^{(1)}, MP^{(2)}, \dots, MP^{(D-1)}, MP^{(D)}\})$$

3. Every possible kDD will manifest in the corresponding kS set. To facilitate their extraction, TSADIS aggregates each discord-score set S by taking the minimum value of its $MP^{(d)}$ elements at each time step. This results in the creation of a Min Matrix Profile (MMP) for each S: $MMP = (\min(S_1), \min(S_2), \dots, \min(S_{L-m+1}))$ where $\min(S_l) = \min_{d \in K}(MP_l^{(d)})$. The min operator ensures that high values MMP values represent timesteps where all the $MP^{(d)}$ are high.
4. An MMP's peaks indicate the locations of the KDDs that occur in its corresponding dimensions. To represent all the possible KDDs in a given KS, TSADIS creates a KD-Profile vector (KDP) that aggregates its MPP discord score values by taking their maximum value at each time step. Formally, $KDP = (\max(MMP_1), \max(MMP_2), \dots, \max(MMP_{L-m+1}))$ where $\max(MMP_l) = \max_d(MMP_l^{(d)})$. In addition, a KD-Profile Index vector ($KDPI$) keeps track of the dimensions that contribute to the KDP at each time step. Finally, TSADIS gathers all KDPs into an All-KD-Profiles ($KDPs$) matrix and all $KDPI$s into an All-KD-Profile Index ($KDPIs$) matrix.
5. The final discord score of a potential kDD at timestep i can be computed by aggregating its corresponding scores from $KDPs$ (in this work, we use a `mean` strategy to penalize each KDD by the number of dimensions on which it occurs, i.e. $\texttt{score}(KDD) = \frac{1}{k} \sum_{d=0}^{k} KDPs_i^{(d)}$), and the k dimensions that contribute to it can be extracted from $KDPIs_i^{(k)}$. The most important kDD is characterized by the highest score.

While being necessary to understand the algorithm, this naive version of TSADIS is highly time consuming. Thus, the authors presented a simple strategy to speed it up. Given that the dicords scores are independent at each time step, the $KDPs$ can be obtained by computing all the MPs, sorting their values at each time step i, and assigning the k-largest value at each i to kDP_i. Following this approach, the KDPs can be computed in $\mathcal{O}(D \log D)$.

4 Proposed approach

4.1 Rationale

The proximity and sparsity properties constrain the magnitude of the perturbation Δ. However, for a counterfactual explanation with target class $C_{m'}$ to be plausible, it has to be similar to other dataset instances from the same class $\mathcal{D}_{m'} = \{\forall T_i | T_i \in C_{m'}\}$. In this section, we introduce DiscoX, a novel model agnostic counterfactual explanation model for univariate and multivariate time series that generates a counterfactual instance by removing the shape descriptors in the original time series T which set it apart from the elements of $\mathcal{D}_{m'}$ and replacing them with their closest subsequences from $\mathcal{D}_{m'}$. This simple mechanism ensures that the counterfactual instance resembles the target class elements and that the perturbation Δ remains small in magnitude. Moreover, by considering all elements of $\mathcal{D}_{m'}$ in the perturbation process, DiscoX generates instances that are close to the entire target class $C_{m'}$ space, increasing its chances to find valid counterfactual explanations.

4.2 Counterfactual explanation generation

Algorithm 1 describes the DiscoX counterfactual explanation generation process for both univariate and multivariate time series. Note that if T is univariate, k and ds will always be equal to 0. The only hyperparameter in DiscoX is the window size m.

First, DiscoX starts by constructing the reference time series $T_{m'}$ by concatenating all the target class instances $\{T_i | T_i \in C_{m'}\}$ with the introduction of a null separator (null array of size D in the multivariate case) between every two instances to prevent the introduction of fabricated patterns that do not exist in the original dataset (Algorithm 1, lines 2–5). Then, it computes all dicords scores and all discords indices matrices DS and DI as described in Algorithm 2. If T is univariate, DS is set to $MP(T, T_{m'})$ and DI to $MPI(T, T_{m'})$ (Algorithm 2, line 2). Otherwise, $KDPs$ and $KDPIs$ are computed using TSADIS and used to compute DI and DS in such a way that each $DI_{d,i}$ element indicates the dimensions that contribute to the discord at position (d, i) and each $DIS_{d,i}$ element stores the score of the discord at position (d, i) (Algorithm 2, lines 4–9). Next, DiscoX creates the target time series instance and initializes it to the original time series (Algorithm 1, line 9). Then, it performs the following steps until a counterfactual explanation is found, i.e. $f(T_{cf}) = C_{m'}$, or until all T subsequences have been completely changed into $T_{m'}$ subsequences[1]:

[1] While not all subsequences are discords, replacing those with low DS values will not have a significant impact since they are highly similar to their nearest neighbors in $T_{m'}$; we chose to follow this strategy instead of introducing a discord threshold hyperparameter value.

Algorithm 1 DISCOX

Inputs: Original time series instance T, dataset $\boldsymbol{\mathcal{D}}$, target class $C_{m'}$, window size m, black-box classifier prediction function f.
Output: Counterfactual explanation T_{cf}.

1: Construct reference time series $T_{m'}$:
2: $T_{m'} \leftarrow$ empty list.
3: **for** T_i in $\boldsymbol{\mathcal{D}}$ **do**
4: **if** $T_i \in C_{m'}$ **then** $T_{m'}.append(T_i, \texttt{NULL})$
5: **end for**
6: Get all discords scores and all discords indices:
7: $DS, DI \leftarrow \texttt{GET\_SCORE}(T, T_{m'}, m)$
8: Expand perturbation until a counterfactual is found:
9: $T_{tg} \leftarrow T$
10: **while** $f(T_{cf}) \neq C_{m'}$ **do**
11: $ds, i, nn\_i, k \leftarrow \texttt{GET\_DISCORD\_NN}(DS, DI, T,$
12: $T_{m'}, m))$
13: $DS[k, i] \leftarrow -\infty$
14: $T_{tg}[d, i : i + m] \leftarrow T_{m'}[d, nn_i : nn_i + m]$
15: **for** $w \leftarrow 0.01$ to 1 **do**
16: $T_{cf} \leftarrow (1 - w) \times T + w \times T_{tg}$
17: **if** $f(T_{cf}) == C_{m'}$ **then** break
18: **end for**
19: **end while**
20: **return** T_{cf}

1. The dimensions and time steps indices *ds* and *i* of the discord with the highest score and its nearest neighbor subsequence location *nn_i* in $T_{m'}$ are extracted as described in Algorithm 3. If *T* is univariate, $DS = MP$ and $DI = MPI$. Therefore, *i* is the location of the highest value in *DS*: $i = \texttt{argmax}(DS)$ and *nn_i* can be directly extracted from *DI* (Algorithm 3, lines 3–4). In case *T* is multivariate, the *i* and *d* indices can also be extracted directly from *DS* and *DI* (Algorithm 3, lines 6–7). However, extracting *nn_i* requires an extra step. Using the Mass similarity search algorithm (the cornerstone of the *MP*) introduced by Yeh et al. (2017b), DiscoX computes the *DP* vectors of $sub^{T}_{m,i}$ and $T_{m'}$ at each dimension in *ds*, sums them up at each time step, and extracts *nn_i* as the time step index with the highest value (Algorithm 3, lines 8–13). Finally, the *DS* value corresponding to the current discord is set to $-\infty$ to exclude it from the next iterations.
2. The subsequence of T_{tg} that is located at the same time steps of the current discord $sub^{T_{tg}}_{i,m}$ is replaced by the discord's nearest neighbor subsequence from the target class instances $sub^{T_{m'}}_{nn_i,m}$ at each dimension in *ds* (Algorithm 1, line 14).
3. The counterfactual explanation T_{cf} is constructed using simple weighted linear interpolation of *T* and T_{tg}. First, DiscoX assigns a higher weight to *T*. Then, it keeps reducing it as long as $f(T_{cf}) \neq C_{m'}$, until the current and previously discovered discords are completely replaced by their closest subsequences

from $T_{m'}$ (Algorithm 1, lines 15–17). In this paper, we use weight increments of $w_{inc} = 10^{-2}$. Smaller increments could result in even smaller Δ, but at the expense of higher computational time.

While it is not the most intuitive way to keep track of discovered discords and of their nearest neighbor subsequences, using T_{tg} allows DiscoX to speed-up the perturbation step (line 16) from $\mathcal{O}(n)$ where n is the number of matched discords to $\mathcal{O}(1)$, by interpolating all discovered discords simultaneously.

4.3 Time complexity

The time complexity of the inner loop in Algorithm 1, line 15 is $\mathcal{O}(1/w_{inc})$ The complexity of the outer loop in Algorithm 1, line 10 is smaller than $\mathcal{O}(L)$ ($\mathcal{O}(L)$ means that an entire iteration is spent to match each single time step).

Let S be the length of $T_{m'}$. If class $C_{m'}$ contains $N_{m'}$ time series instances, $S = N_{m'} \times (L' + 1) - 1$.

4.3.1 Univariate time series ($D = 0$)

The worst case time complexity of `GET_SCORE()` is $\mathcal{O}(S^2)$, the complexity of STOMP. The time complexity of `GET_DISCORD_NN()` is $\mathcal{O}(L)$, the complexity of `argmax`. Therefore, the complexity of the entire perturbation procedure (lines 10–19) –and thus of DiscoX– is smaller than $\mathcal{O}(LS^2)$.

4.3.2 Multivariate time series ($D > 0$)

The worst case time complexity of `GET_SCORE()` is $\mathcal{O}(D \log D)$, the complexity of TSADIS. Since the complexity of the Mass algorithm is $\mathcal{O}(SlogS)$, the time complexity of `GET_DISCORD_NN()` is $\mathcal{O}(DSlog(L_{m'}))$. Therefore, the complexity of the entire perturbation procedure (lines 10–19) –and thus of DiscoX– is smaller than $\mathcal{O}(DLSlog(S))$. Note here that the loop in Algorithm 3, line 9 is embarrassingly parallel. Therefore, the time complexity can be reduced to $\mathcal{O}(LSlog(S))$ in the presence of sufficient computational cores.

Algorithm 2 `GET_SCORES()`

Inputs: Query time series instance T, reference time series $T_{m'}$, window size m
Output: All discords scores matrix DS, all discords indices matrix DI.

if T is univariate (i.e. $T \in \mathbb{R}^L$) **then**
  $DS, DI \leftarrow STOMP(T, T_{m'}, m)$ ▷ **(Zhu et al., 2018)**
else if T is multivariate (i.e. $T \in \mathbb{R}^{D \times L}$) **then**
  $KDPs, KDPIs \leftarrow \texttt{TSADIS}(T, T_{m'}[d], m)$
  $DS, DI \leftarrow$ arrays of size $D \times L$
  for $d \leftarrow 0$ to D **do**
    $s \leftarrow KDPs[: d+1]$
    $DI[d] \leftarrow KDPIs[: d+1]$
    $DS[d] \leftarrow \texttt{mean}(s[: d+1], axis = 0)$
  end for
end if
return DS, DI

Algorithm 3 `GET_DISCORD_NN()`

Inputs: All discords scores matrix DS, all discords indices matrix DI, query time series instance T, reference time series $T_{m'}$, window size m
Output: Top discord dimensions ds and time step i, nearest neighbor subsequence time step nn_i, top $KDPs$ dimension k.

if DS is univariate **then**
  $ds \leftarrow 1$
  $i \leftarrow \texttt{argmax}(DS)$
  $nn\_i \leftarrow DI[i]$
else if DS is multivariate **then**
  $k, i \leftarrow \texttt{argmax}(DS)$
  $ds \leftarrow DI[k, i]$
  $DPs \leftarrow$ arrays of size $\texttt{len}(ds) \times L - m + 1$
  for d in ds **do**
    $DP[d] \leftarrow \texttt{MASS}(T[d, i : i + m], T_{m'}, m)$ ▷ **Yeh et al. (2017)**
  end for
  $DP \leftarrow \texttt{sum}(DP, axis = 0)$
  $nn\_i \leftarrow \texttt{argmax}(DP)$
end if
return ds, i, nn_i, k

Table 1 Datasets details

Dataset	Train Size	Test Size	Classes	Length	Dimensions
BME	30	150	3	128	1
CBF	30	900	3	128	1
Coffee	28	28	2	286	1
GunPoint	150	150	2	150	1
GunPointAgeSpan	135	316	2	150	1
ECGFiveDays	23	861	2	136	1
InsectEPGRegularTrain	62	149	3	601	1
Meat	60	60	3	448	1
Plane	105	105	7	144	1
ShapeletSim	20	180	2	500	1
Trace	100	100	4	275	1
ArticularyWordRecognition	275	300	25	144	9
BasicMotions	40	40	4	100	6
Cricket	108	72	1197	12	6
Epilepsy	137	138	4	206	3

5 Experimental setup

5.1 Black-box classification models

We conduct our experiments using two different time series black-box classification models: Residual Network (ResNet) (Wang et al. 2017) and RandOm Convolutional KErnel Transform (MrSQM) (Nguyen and Ifrim 2023). The former is a classic deep learning architecture for time series classification that achieved good results on the UCR archive in a recent benchmark (Ismail Fawaz et al. 2019). The latter is a symbolic representation based model that achieves state-of-the art results with a very short runtime. We keep default parameter configurations for both architectures and train ResNet for 1500 epochs, similarly to Ismail Fawaz et al. (2019).

5.2 Datasets

We evaluate the counterfactual instances generated from the UCR archive (Dau et al. 2018) of univariate time series datasets and the UEA archive (Hoang et al. 2018) of multivariate time series datasets. First, we train ResNet and MrSQM models on the proposed training sets of all 117 fixed-length datasets from UCR (we discard the 11 variable-length ones to avoid preprocessing issues) and all 26 datasets from UEA. Then, we test the models' classification performances on their respective test sets and select the datasets with f1-scores of 100% from the UCR archive to ensure that the counterfactual explanations are robust, i.e. that the label did flip due to uncertainty on the classification model side. For the UEA archive, we lower the f1-score threshold to

95% since none of the datasets yielded a 100% value. This results in 8 and 7 datasets from UCR for ResNet and MrSQM respectively, and 4 datasets from UEA for both models (see Table 1 for details of the datasets).

5.3 Baselines and implementation details

We compare the DiscoX counterfactual explanations to those generated by three baseline methods.

1. **CEGP:** (Van Looveren and Klaise 2019) an optimization based algorithm that improves the one by Wachter et al. (Wachter et al. 2018). CEGP was originally proposed for tabular and image data. However, it can be easily adapted for time series. We use it as the representative algorithm for optimization-based explanation models instead of CEM (Dhurandhar et al. 2018) since the latter requires training autoencoders models for each dataset, while CEGP provides the option of replacing them by KD-tree class prototypes.

   CEGP is originally designed for neural network black-box classifiers, as it needs the f prediction function to be differentiable to compute its gradients. The authors also proposed an algorithm to compute the gradients for other types of models. However, the runtime is so slow that they chose not to include it in their experiments. Since our experiments using MrSQM proved similar, we only use CEGP to explain ResNet predictions.
2. **Native Guide (NG):** (Delaney et al. 2020) in case the black-box model is differentiable, NG introduces perturbations at the most important time steps of the original instance, extracted from the Class Activation Map (Zhou et al. 2015) of the black-box neural network model, by injecting values from the target class nun at the same time steps. Otherwise, it uses DBA (Petitjean et al. 2011) with incremental weights to generate the counterfactual instance by interpolating the original time series and the reference nun.
3. **CoMTE:** (Ates et al. 2021) is the first counterfactual explanation method designed specifically for multivariate time series. It works by selecting a nun from the target class and replacing entire dimensions in the original time series with dimensions from the nun.

The source code for DiscoX is available in our project website.[2] The only hyperparameter of DiscoX is the window size m. Ideally, it should be defined by domain experts for each different dataset. In the following experiments, we set it to $m = 0.1 \times L$. Since NG and CoMTE default to the nun in case they fail to find a counterfactual instance, we follow the same approach for the sake of proximity and sparsity evaluation only. However, since the nuns belong to the original data it does not make sense to include them in the plausibility evaluation.

[2] https://sites.google.com/view/DiscoX/home.

Table 2 Comparing proximity scores for counterfactual explanations generated from univariate UCR datasets using ResNet as the black-box classifier

	L1			L2		
Dataset	NG	CEGP	DiscoX	NG	CEGP	DiscoX
BME	**25.79**	27.06	25.97	**3.61**	4.32	3.80
Coffee	12.92	14.58	**8.66**	**1.10**	1.93	1.26
GunPoint	15.24	**4.96**	6.03	1.87	**0.91**	1.08
InsectEPGRegularTrain	631.67	1,335.12	**541.74**	**26.57**	66.41	29.51
Meat	6.70	16.80	**6.46**	**0.46**	1.42	0.55
Plane	72.94	43.21	**31.78**	8.04	5.04	**4.72**
ShapeletSim	276.96	**18.33**	114.87	15.86	**2.12**	10.02
Trace	165.46	326.38	**36.36**	13.92	25.97	**5.56**
Average Rank	2.25	2.38	**1.38**	1.88	2.38	**1.75**

For each dataset, the bold highlights the method with the best performance

Table 3 Comparing proximity scores for counterfactual explanations generated from univariate UCR datasets using MrSQM as the black-box classifier

	L1		L2	
Dataset	NG	DiscoX	NG	DiscoX
CBF	67.69	**32.23**	7.85	**4.89**
Coffee	15.36	**11.37**	**1.32**	1.41
GunPointAgeSpan	5366.00	**1143.38**	635.93	**120.52**
ECGFiveDays	7.96	**7.50**	**1.23**	1.47
Plane	59.65	**38.33**	6.53	**5.44**
ShapeletSim	353.31	**212.88**	21.42	**14.19**
Trace	111.70	**65.24**	9.66	**7.81**
Average Rank	2	**1**	1.86	**1.14**

For each dataset, the bold highlights the method with the best performance

6 Experimental results

We generate counterfactual explanation instances using DiscoX, NG, CEGP, and CoMTE to explain the predictions of the ResNet and MrSQM classification models trained on the selected datasets. For each time series instance T from the test set with class prediction $f(T) = C_m$, we produce a counterfactual explanation with each of the other dataset classes set as the target class $C_{m'} \in \mathcal{C} - C_m$.

Table 4 Comparing proximity scores for counterfactual explanations generated from multivariate UEA datasets using ResNet as the black-box classifier

	L1		L2	
Dataset	COMTE	DiscoX	COMTE	DiscoX
Articulary-WordRecognition	1,474.69	**652.15**	50.69	**28.09**
BasicMotions	2,229.03	**1178.75**	133.51	**89.35**
Cricket	7,819.35	**3814.10**	119.59	**75.27**
Epilepsy	507.21	**351.90**	25.83	**21.02**
Average Rank	2	**1**	2	**1**

For each dataset, the bold highlights the method with the best performance

Table 5 Comparing proximity scores for counterfactual explanations generated from multivariate UEA datasets using MrSQM as the black-box classifier

	L1		L2	
Dataset	COMTE	DiscoX	COMTE	DiscoX
Articulary-WordRecognition	1475.39	**1125.33**	50.69	**40.26**
BasicMotions	2204.98	**1542.49**	132.13	**100.31**
Cricket	7821.33	**4957.36**	119.58	**86.08**
Epilepsy	552.07	**331.08**	28.07	**19.10**
Average Rank	2	**1**	2	**1**

For each dataset, the bold highlights the method with the best performance

Therefore, for a dataset with N time series instances and M number of classes, we attempt to create $N \times M$ counterfactual explanations.

6.1 Proximity

We adopt the L_1- and L_2–norms as proximity measures to compare how small are the perturbations Δ introduced by each method. We compute $||\Delta||_1$ and $||\Delta||_2$ for the explanations of each dataset and display the results in addition to the average ranks over all datasets in Tables 2 and 3 for the UCR datasets and Tables 4 and 5 for the UEA datasets. The results show that DiscoX ranks consistently better compared to the other baselines using both metrics. This means that DiscoX generates counterfactual instances that are closer to the original time series.

6.2 Plausibility

We use the local outlier factor (LOF) novelty detection algorithm Breunig et al. (2000); Kanamori et al. (2020), which computes the local density variation for each instance in relation to its neighbors and identifies instances with lower

Table 6 Comparing plausibility, sparsity, and validity scores for counterfactual explanations generated from univariate UCR datasets using ResNet as the black-box classifier

	LOF			Sparsity			Failure Rate		
Dataset	NG	CEGP	DiscoX	NG	CEGP	DiscoX	NG	CEGP	DiscoX
BME	**0.01**	0.43	0.23	**0.67**	1.00	0.78	0.67	**0.33**	0.50
Coffee	0.31	0.9	**0.08**	0.54	1.00	**0.28**	0.54	0.21	**0.00**
GunPoint	0.5	**0.22**	0.3	0.51	1.00	**0.32**	0.51	0.14	**0.01**
InsectEPGRegularTrain	1	1	1	**0.48**	1.00	0.67	0.68	0.12	**0.00**
Meat	0.88	0.87	**0.34**	0.68	1.00	**0.58**	0.67	0.33	**0.03**
Plane	**0.92**	0.96	0.98	0.67	1.00	**0.57**	0.86	0.32	**0.05**
ShapeletSim	**0**	**0**	**0**	0.86	1.00	**0.56**	0.50	**0.00**	0.02
Trace	**0.62**	0.81	0.7	0.50	1.00	**0.45**	0.75	0.23	**0.06**
Average Rank	**1.63**	2	**1.63**	1.78	3.00	**1.22**	3	1.75	**1.25**

For each dataset, the bold highlights the method with the best performance

Table 7 Comparing plausibility, sparsity, and validity scores for counterfactual explanations generated from univariate UCR datasets using MrSQM as the black-box classifier

	LOF		Sparsity		Failure Rate	
Dataset	NG	DiscoX	NG	DiscoX	NG	DiscoX
CBF	**0**	0.18	0.67	**0.44**	0.67	**0.06**
Coffee	12.92	**8.66**	0.89	**0.38**	0.25	**0**
GunPointAgeSpan	0.57	**0.53**	0.51	**0.36**	0.51	**0.21**
ECGFiveDays	**0.08**	0.15	0.51	**0.46**	0.50	**0**
Plane	**0.92**	0.98	0.97	**0.63**	0.41	**0.07**
ShapeletSim	**0**	**0**	0.85	**0.59**	0.32	**0.21**
Trace	**0.61**	0.69	0.99	**0.51**	**0.08**	0.15
Average Rank	**1.29**	1.71	2	**1**	1.71	**1.29**

For each dataset, the bold highlights the method with the best performance

Table 8 Comparing plausibility, sparsity, and validity scores for counterfactual explanations generated from multivariate UEA datasets using ResNet as the black-box classifier

	LOF		Sparsity		Failure Rate	
Dataset	COMTE	DiscoX	COMTE	DiscoX	COMTE	DiscoX
Articulary-WordRecognition	0.96	**0.72**	1.00	**0.66**	1.00	**0.14**
BasicMotions	0.5	**0.4**	1.00	**0.62**	1.00	**0.07**
Cricket	0.59	**0.35**	1.00	**0.58**	1.00	**0.07**
Epilepsy	0.36	**0.31**	0.99	**0.73**	1.00	**0.14**
Average Rank	2	**1**	2	**1**	2	**1**

For each dataset, the bold highlights the method with the best performance

Table 9 Comparing plausibility, sparsity, and validity scores for counterfactual explanations generated from multivariate UEA datasets using MrSQM as the black-box classifier

	LOF		Sparsity		Failure Rate	
Dataset	CoMTE	DiscoX	CoMTE	DiscoX	CoMTE	DiscoX
Articulary-WordRecognition	0.96	**0.55**	1.00	**0.95**	1.00	**0.85**
BasicMotions	0.5	**0.42**	1.00	**0.76**	1.00	**0.5**
Cricket	0.59	**0.11**	1.00	**0.83**	1.00	**0.72**
Epilepsy	0.36	**0.32**	1.00	**0.77**	1.00	**0.55**
Average Rank	2	**1**	2	**1**	2	**1**

For each dataset, the bold highlights the method with the best performance

densities as outliers, to evaluate the plausibility of each counterfactual method explanations. In Tables 6 and 7, we display the outlier factor values (ratio of counterfactual explanations detected as outliers compared to their target class instances) of each method over for the UCR datasets. in addition to the average ranks. The results for the UEA datasets are displayed in Tables 8 and 9. Using ResNet as the black-box classifier, the univariate explanations generated by DiscoX and NG had the same proportion of outliers. On the other hand, NG generate significantly fewer outliers with MrSQM as the black-box classifier. Therefore, the DBA version of NG does better in this regard compared to the original CAM version. CEGP and CoMTE generated more outliers in all cases.

6.3 Sparsity

We measure sparsity as the total number of time steps that have been modified in the original time series to create the counterfactual instance, i.e. the sum of non-zero elements in the perturbation vector $||\Delta||_0$. DiscoX is the clear winner when it comes to this criteria, with a little competition from NG for the UCR datasets using ResNet only. With MrSQSM as the black-box model, NG introduces perturbations at almost every time step as expected results from its DBA version (the values are not equal to 1.0 since some nun time steps are similar to the original ones). As to CEGP and CoMTE, they both modifiy every single time step for all datasets.

6.4 Validity

Finally, we assess the validity of each method by comparing the failure rate on each dataset, i.e. the ratio of the total number of times where the method failed to find a counterfactual explanation over the total number of attempts ($N \times M$). Once more, DiscoX was the most successful in finding explanations, this time followed by CEGP. NG had a significantly higher failure rate, particularly using

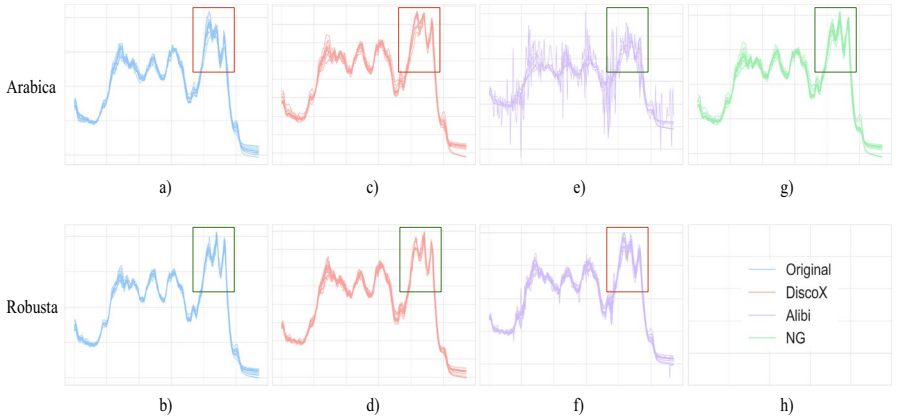

Fig. 1 Comparing Coffee counterfactual explanation instances generated using DiscoX (**c**, **d**), CEGP (**e**, **f**), and NG (**g**) to the original training set time series (**a**, **b**). Legend provided in (**h**) (Color figure online)

the prioritized CAM version where it did not go below 50%. As to CoMTE, it appears that it has defaulted to the nun for all instances.

6.5 Visual analysis

In this section, we evaluate the counterfactual instances by comparing their plots to the original data and by projecting their distributions on a 2-D space. For this purpose, we select two food spectrography datasets from Table 1, namely Coffee and Meat, as they both contain well separated and visually intelligible classes. The Coffee dataset was first introduced by Briandet et al. (1996) for the task of distinguishing between Arabica and Robusta coffee beans. The Meat dataset first appeared in a paper by Al-Jowder et al. (1997). It contains spectrographs of three types of meat: Chicken, Turkey, and Pork. We consider the explanations generated using the ResNet model, as both NG and CEGP favor neural network classifiers. We limit this study to univariate datasets since (1) they are easier to visualize and (2) CoMTE, the only multivariate specific algortihm defaulted to the nuns for all datasets instances.

6.5.1 Visualizing counterfactual explanations

In Figs. 1 and 2, we compare the original spectrographs to the generated ones. Each row contains spectrogaphs from a given class (either belong to it or predicted by ResNet). The first column contains the original spectrogaphs from the training sets. The second, third, and fourth columns contain the intances generated by DiscoX, CEGP, and NG respectively. In each subfigure, we plot all the timeseries instances belonging to the particular "class-algorithm" pair (or "class-training set" for the first column).

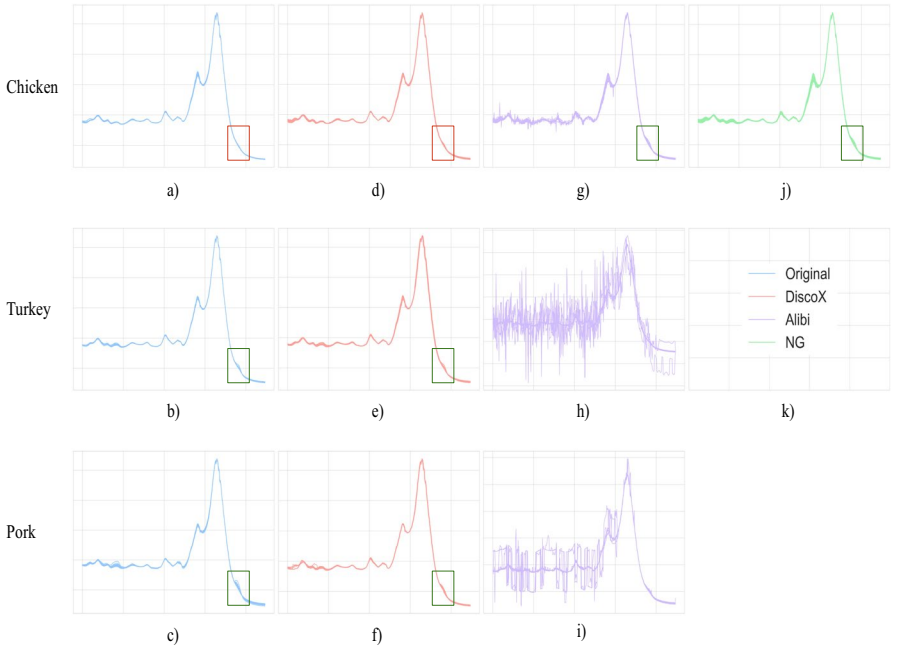

Fig. 2 Comparing Meat counterfactual explanation instances generated using DiscoX (**d**–**f**), CEGP (**g**–**i**), and NG (**j**) to the original training set time series (**a**–**c**). Legend provided in (**k**) (Color figure online)

Coffee Dataset The original spectrographs in Fig. 1a, b show the existence of a main characteristic feature that differentiates Arabica beans from Robusta beans, located in the rectangular boxes. For the Arabica class, the three peaks are aligned along a descending diagonal line \ (framed in red), whereas for the Robusta class, the three peaks from a reversed V shape Λ with the highest one in the middle (framed in green).

By looking at the DiscoX spectrographs, it is clear that only DiscoX was able to transform the diagonal shape \ into the reversed V shape Λ when creating counterfactual instances for the Robusta class time series, and vice-versa. On the other hand, the counterfactual instances generated by CEGP and NG preserved the characteristic feature of their original classes (Fig. 1e–g). This makes them visually similar to the original classes whereas, in theory, they should be similar to the target class. For example, the Robusta spectographs in Fig. 1f resemble the original Arabica ones in Fig. 1a and the Arabica spectorgraphs in Fig. 1g resemble the original Robusta ones in Fig. 1b. In addition, the CEGP optimization process results in noisy, unrealistic time series that contain oscillations throughout their length. Moreover, NG was unable to generate counterfactuals with Robusta as a target class.

Meat Dataset As pointed to in the original paper (Al-Jowder et al. 1997), the original spectrographs in Fig. 2a–c contain a main characteristic feature that differentiates the three types of meats as show in the rectangular boxes. In particular, the Chicken spectrographs in that area are defined by a smooth U-curve (framed in

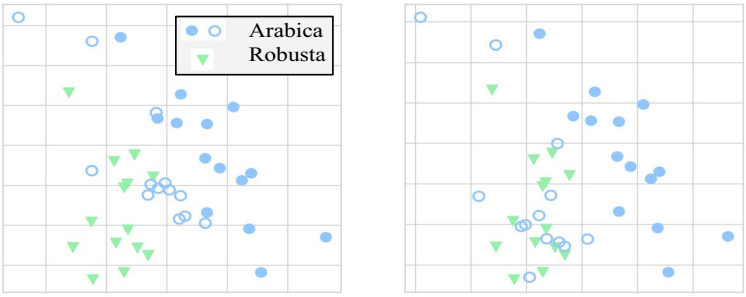

Fig. 3 Comparing the distributions (first two PCA components) of Coffee counterfactual explanation instances generated using DiscoX (**a**) and NG (**b**) to the original training set time series

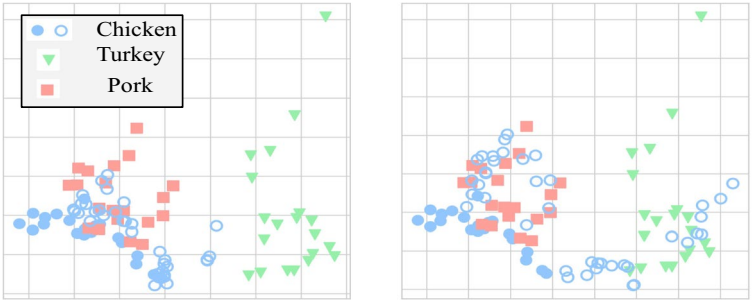

Fig. 4 Comparing the distributions (first two PCA components) of Meat counterfactual explanation instances generated using DiscoX (**a**) and NG (**b**) to the original training set time series

red), whereas the Turkey and, more noticeably, Pork spectographs contain a small bump (framed in green).

Similarly to the Coffee dataset, only DiscoX was able to smoothen the bump when generating Chicken counterfactuals for Turkey and Pork instances (Fig. 2d). On the other hand, CEGP and NG failed in eliminating this characteristic feature as shown in the green boxes in Fig. 2g, j. In addition, the instances generated by CEGP suffer from the same oscillations problem and NG only succeeded in finding explanations when Chicken is set as a target class.

From the above observations, we conclude that DiscoX introduces meaningful perturbations. Therefore, the explanations it provides can be easily interpretable and understood by stakeholders. Also, we note that (1) CEGP's oscillation problem explains its low plausibiliy performance in Sect. 6.2 and (2) NG's success with one target class explains its low validity rate in Sect. 6.4.

6.5.2 Principal component analysis

In this section, we use Principal Component Analysis (PCA) to visualize the distributions of the counterfactual explanation samples by projecting them onto the 2-D space defined by the first two principal components of the training set. We discard the CEGP counterfactuals since they proved to be unrealistic in the previous section and visualize the counterfactuals from the classes where NG succeeded only. In Figs. 3 and 4, each class is represented by a combination of color and shape, with the unfilled shapes representing original training data samples and the filled shapes representing the counterfactual data.

Figures 3b and 4b show that most of the counterfactual instances generated by NG are in the vicinity of original instances from the opposite class: apart from a few exceptions, the Arabica counterfactuals are close to the Robusta original instances and the Chicken counterfactuals are spread around the Turkey and Pork original instances. This explains the fact that NG preserved the characteristic features of the original timeseries when generating counterfactuals in the previous section. On the other hand, Fig. 4a, b show that DiscoX generated most of the counterfactual instances in the border regions between the different classes, which precisely illustrates the proximity-plausibility trade-off.

7 Conclusion and future work

We proposed DiscoX, a posthoc, model agnostic counterfactual explanation algorithm for univariate and multivariate time series classification models. DiscoX utilizes the Matrix Profile (MP) all-pair similarity search to detect discords in the original time series with regards to all the target class data instances and to replace them with their closest matches. Using real-life datasets from the UCR and UEA archives, we compare the performance of DiscoX to three state-of-the-art counterfactual explanation methods. The results show that our algorithm produces the best counterfactuals in terms of proximity, sparsity, and validity. Moreover, we prove that DiscoX introduces more meaningful perturbations. By visualizing the counterfactual explanations created by DiscoX, we show that it is able to introduce human-understandable changes to the original data. The next step would be to verify these results and to experiment by conducting a user study with domain experts. In addition, we would like to test the ability of DiscoX to create explanations according to different domain-specific goals. For example, by allowing users to select target discords.

Acknowledgements This project has been supported in part by funding from GEO Directorate under NSF awards #2204363, #2240022, and #2301397 and the CISE Directorate under NSF award #2305781.

References

Al-Jowder O, Kemsley EK, Wilson RH (1997) Mid-infrared spectroscopy and authenticity problems in selected meats: a feasibility study. Food Chem 59:195–201. https://doi.org/10.1016/S0308-8146(96)00289-0

Ates E, Aksar B, Leung VJ, Coskun AK (2021) Counterfactual explanations for multivariate time series. In: International conference on applied artificial intelligence (ICAPAI), pp 1–8. https://doi.org/10.1109/ICAPAI49758.2021.9462056

Bahri O, Boubrahimi SF, Hamdi SM (2022a) Shapelet-based counterfactual explanations for multivariate time series. In: ACM SIGKDD workshop on mining and learning from time series (KDD-MiLeTS 2022)

Bahri O, Li P, Filali Boubrahimi S, Hamdi SM (2022b) Temporal rule-based counterfactual explanations for multivariate time series. In: 2022 21st IEEE international conference on machine learning and applications (ICMLA), Nassau, Bahamas, pp 1244–1249. https://doi.org/10.1109/ICMLA55696.2022.00200

Breunig MM, Kriegel HP, Ng RT, Sander J (2000) LOF: identifying density-based local outliers. ACM SIGMOD Rec 29(2):93–104. https://doi.org/10.1145/335191.335388

Briandet R, Kemsley EK, Wilson RH (1996) Discrimination of arabica and Robusta in instant coffee by Fourier transform infrared spectroscopy and chemometrics. J Agric Food Chem 44:170–174. https://doi.org/10.1021/JF950305A

Dau HA, Bagnall A, Kamgar K, Yeh C-CM, Zhu Y, Gharghabi S, Ratanamahatana CA, Keogh E (2018) The UCR time series archive. arXiv:1810.07758

Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6:182–197. https://doi.org/10.1109/4235.996017

Delaney E, Greene D, Keane MT (2020) Instance-based counterfactual explanations for time series classification. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics) 12877 LNAI, pp 32–47. https://doi.org/10.1007/978-3-030-86957-1_3. arXiv:2009.13211

Delaney E, Pakrashi A, Greene D, Keane MT (2023) Counterfactual explanations for misclassified images: how human and machine explanations differ. Artif Intell 324:103995. https://doi.org/10.1016/J.ARTINT.2023.103995

Dhurandhar A, Chen PY, Luss R, Tu CC, Ting P, Shanmugam K, Das P (2018) Explanations based on the missing: towards contrastive explanations with pertinent negatives. In: Advances in neural information processing systems, pp 592–603. https://doi.org/10.48550/arxiv.1802.07623arXiv:1802.07623

Filali Boubrahimi S, Hamdi SM (2022) On the mining of time series data counterfactual explanations using Barycenters. In: International conference on information and knowledge management, proceedings, pp 3943–3947. Association for Computing Machinery, New York. https://doi.org/10.1145/3511808.3557663

Gharghabi S, Imani S, Bagnall A, Darvishzadeh A, Keogh E (2018) Matrix Profile XII: MPdist: a novel time series distance measure to allow data mining in more challenging scenarios. In: Proceedings—IEEE international conference on data mining, ICDM, pp 965–970. https://doi.org/10.1109/ICDM.2018.00119

Guidotti R (2022) Counterfactual explanations and how to find them: literature review and benchmarking. Data Min Knowl Discov. https://doi.org/10.1007/S10618-022-00831-6/FIGURES/4

Hoang AB, Dau A, Lines J, Flynn M, Large J, Bostrom A, Southam P, Keogh E (2018) The UEA multivariate time series classification archive, 2018. arXiv:1811.00075v1

Höllig J, Kulbach C, Thoma S (2022) TSEvo: Evolutionary counterfactual explanations for time series classification. In: Proceedings—21st IEEE international conference on machine learning and applications, ICMLA 2022, pp 29–36 (2022). https://doi.org/10.1109/ICMLA55696.2022.00013

Ismail Fawaz H, Forestier G, Weber J, Idoumghar L, Muller PA (2019) Deep learning for time series classification: a review. Data Min Knowl Discov 33(4):917–963. https://doi.org/10.1007/s10618-019-00619-1. arXiv:1809.04356

Kanamori K, Takagi T, Kobayashi K, Arimura H (2020) DACE: distribution-aware counterfactual explanation by mixed-integer linear optimization. In: IJCAI international joint conference on artificial intelligence, vol 3, pp 2855–2862. https://doi.org/10.24963/IJCAI.2020/395

Karimi A-H, Barthe G, Balle B, Valera I (2020) Model-agnostic counterfactual explanations for consequential decisions. In: PMLR (ed.) International conference on artificial intelligence and statistics

Karlsson I, Papapetrou P, Boström H (2016) Generalized random shapelet forests. Data Min Knowl Disc 30(5):1053–1085. https://doi.org/10.1007/S10618-016-0473-Y

Karlsson I, Rebane J, Papapetrou P, Gionis A (2020) Locally and globally explainable time series tweaking. Knowl Inf Syst 62:1671–1700. https://doi.org/10.1007/S10115-019-01389-4/FIGURES/14

Keane MT, Kenny EM, Delaney E, Smyth B (2021) If only we had better counterfactual explanations: five key deficits to rectify in the evaluation of counterfactual XAI techniques. In: IJCAI international joint conference on artificial intelligence, pp 4466–4474. https://doi.org/10.48550/arxiv.2103.01035

Labaien J, Zugasti E, Carlos XD (2020) Contrastive explanations for a deep learning model on time-series data. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics) 12393 LNCS, pp 235–244. https://doi.org/10.1007/978-3-030-59065-9_19

Li P, Bahri O, Boubrahimi SF, Hamdi SM (2022a) Sg-cf: Shapelet-guided counterfactual explanation for time series classification, pp 1564–1569. Institute of Electrical and Electronics Engineers Inc., Osaka, Japan. https://doi.org/10.1109/BigData55660.2022.10020866

Li P, Boubrahimi SF, Hamdi SM (2022b) Motif-guided time series counterfactual explanations. In: 2nd workshop on explainable and ethical AI-ICPR 2022. arXiv:2211.04411

Liu FT, Ting KM, Zhou ZH (2008) Isolation forest. In: Proceedings—IEEE international conference on data mining, ICDM, pp 413–422. https://doi.org/10.1109/ICDM.2008.17

Looveren AV, Klaise J, Vacanti G, Cobb O (2021) Conditional generative models for counterfactual explanations. arXiv:2101.10123v1

Lu Y, Wu R, Mueen A, Zuluaga MA, Keogh E (2022) Matrix Profile XXIV: scaling time series anomaly detection to trillions of datapoints and ultra-fast arriving data streams. In: Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining, pp 1173–1182. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3534678.3539271

Mothilal RK, Sharma A, Tan C (2020) Explaining machine learning classifiers through diverse counterfactual explanations. In: Proceedings of the 2020 conference on fairness, accountability, and transparency, Barcelona, Spain. https://doi.org/10.1145/3351095.3372850

Nguyen TL, Ifrim G (2023) Fast time series classification with random symbolic subsequences. In: Advanced analytics and learning on temporal data: 7th ECML PKDD workshop, AALTD 2022, Grenoble, France, September 19–23, 2022, Revised Selected Papers, pp 50–65. Springer, Berlin. https://doi.org/10.1007/978-3-031-24378-3_4

Petitjean F, Ketterlin A, Gançarski P (2011) A global averaging method for dynamic time warping, with applications to clustering. Pattern Recogn 44(3):678–693. https://doi.org/10.1016/J.PATCOG.2010.09.013

Rojat T, Puget R, Filliat D, Ser JD, Gelin R, Díaz-Rodríguez N (2021) Explainable artificial intelligence (XAI) on timeseries data: a survey

Schölkopf B, Platt JC, Shawe-Taylor J, Smola AJ, Williamson RC (2001) Estimating the support of a high-dimensional distribution. Neural Comput 13(7):1443–1471. https://doi.org/10.1162/089976601750264965

Tafazoli S, Keogh E (2023) Matrix profile xxviii: Discovering multi-dimensional time series anomalies with k of n anomaly detection, pp 685–693. Society for Industrial and Applied Mathematics, Philadelphia, PA. https://doi.org/10.1137/1.9781611977653.ch77

Theissler A, Spinnato F, Schlegel U, Guidotti R (2022) Explainable AI for time series classification: a review, taxonomy and research directions. IEEE Access 10:100700–100724. https://doi.org/10.1109/ACCESS.2022.3207765

Van Looveren A, Klaise J (2019) Interpretable counterfactual explanations guided by prototypes. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics) 12976 LNAI, pp 650–665. https://doi.org/10.48550/arxiv.1907.02584arXiv:1907.02584

Wachter S, Mittelstadt B, Russell C (2018) Counterfactual explanations without opening the black box: automated decisions and the GDPR. Harvard J Technol. https://doi.org/10.2139/ssrn.3063289

Wang Z, Yan W, Oates T (2017) Time series classification from scratch with deep neural networks: a strong baseline. In: Proceedings of the international joint conference on neural networks, vol 2017-May, pp 1578–1585. https://doi.org/10.1109/IJCNN.2017.7966039

Yeh CCM, Kavantzas N, Keogh E (2017a) Matrix profile VI: Meaningful multidimensional motif discovery. In: Proceedings—IEEE international conference on data mining, ICDM 2017-November, pp 565–574. https://doi.org/10.1109/ICDM.2017.66

Yeh C-CM, Zhu Y, Ulanova L, Begum N, Ding Y, Dau HA, Silva DF, Mueen A, Keogh E (2017b) Matrix Profile I: all pairs similarity joins for time series: a unifying view that includes Motifs, Discords and Shapelets, pp 1317–1322. https://doi.org/10.1109/ICDM.2016.0179

Zhou B, Khosla A, Lapedriza A, Oliva A, Torralba A (2015) Learning deep features for discriminative localization. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition 2016-Decem, pp 2921–2929 (2015) https://doi.org/10.48550/arxiv.1512.04150arXiv:1512.04150

Zhu Y, Zimmerman Z, Shakibay Senobari N, Yeh CCM, Funning G, Mueen A, Brisk P, Keogh E (2018) Exploiting a novel algorithm and GPUs to break the ten quadrillion pairwise comparisons barrier for time series motifs and joins. Knowl Inf Syst 54:203–236. https://doi.org/10.1007/S10115-017-1138-X

Zhu Y, Gharghabi S, Silva DF, Dau HA, Yeh CCM, Senobari NS, Almaslukh A, Kamgar K, Zimmerman Z, Funning G, Mueen A, Keogh E (2020) The swiss army knife of time series data mining: ten useful things you can do with the matrix profile and ten lines of code. Data Min Knowl Discov 34:949–979. https://doi.org/10.1007/S10618-019-00668-6/METRICS