# Understanding Any Time Series Classifier with a Subsequence-based Explainer

FRANCESCO SPINNATO, Scuola Normale Superiore, Italy and ISTI-CNR, Italy
RICCARDO GUIDOTTI and ANNA MONREALE, University of Pisa, Italy
MIRCO NANNI, ISTI-CNR, Italy
DINO PEDRESCHI, University of Pisa, Italy
FOSCA GIANNOTTI, Scuola Normale Superiore, Italy

The growing availability of time series data has increased the usage of classifiers for this data type. Unfortunately, state-of-the-art time series classifiers are black-box models and, therefore, not usable in critical domains such as healthcare or finance, where explainability can be a crucial requirement. This paper presents a framework to explain the predictions of any black-box classifier for univariate and multivariate time series. The provided explanation is composed of three parts. First, a saliency map highlighting the most important parts of the time series for the classification. Second, an instance-based explanation exemplifies the black-box's decision by providing a set of prototypical and counterfactual time series. Third, a factual and counterfactual rule-based explanation, revealing the reasons for the classification through logical conditions based on subsequences that must, or must not, be contained in the time series. Experiments and benchmarks show that the proposed method provides faithful, meaningful, stable, and interpretable explanations.

CCS Concepts: • **Mathematics of computing** → **Time series analysis**; • **Information systems** → **Data mining**; **Decision support systems**; • **Computing methodologies** → **Supervised learning by classification;**

Additional Key Words and Phrases: Explainable AI, time series classification, subsequence-based rules, prototypes and counterfactuals

This work has been supported by SoBigData.it that receives funding from European Union – NextGenerationEU – National Recovery and Resilience Plan (Piano Nazionale di Ripresa e Resilienza, PNRR) – Project: "SoBigData.it – Strengthening the Italian RI for Social Mining and Big Data Analytics" – Prot. IR0000013 – Avviso n. 3264 del 28/12/2021. The research developed in this works is also aligned with the research partially supported by the European Community Horizon 2020 programme under the funding schemes: H2020-INFRAIA-2019-1: Research Infrastructure G.A. 871042 *SoBigData++*, G.A. 952026 *HumanE-AI Net*, ERC-2018-ADG G.A. 834756 *XAI: Science and technology for the eXplanation of AI decision making*, G.A. 952215 *TAILOR*, and by the NextGenerationEU programme under the funding schemes PNRR-PE-AI scheme (M4C2, investment 1.3, line on AI) *FAIR* (Future Artificial Intelligence Research).

Authors' addresses: F. Spinnato, Scuola Normale Superiore, Piazza dei Cavalieri, 7, Pisa, Italy, 56126, Italy and ISTI-CNR, Via Giuseppe Moruzzi, 1, Pisa, Italy, 56124, Italy; e-mail: francesco.spinnato@sns.it, francesco.spinnato@isti.cnr.it; R. Guidotti, A. Monreale, and D. Pedreschi, University of Pisa, Largo B. Pontecorvo, 3, Pisa, Italy, 56127, Italy, e-mails: riccardo.guidotti@unipi.it, {anna.monreale, dino.pedreschi}@.unipi.it; M. Nanni, ISTI-CNR, Via Giuseppe Moruzzi, 1, Pisa, Italy, 56124, Italy; e-mail: mirco.nanni@isti.cnr.it; F. Giannotti, Scuola Normale Superiore, Piazza dei Cavalieri, 7, Pisa, Italy, 56126, Italy, fosca.giannotti@sns.it.

1556-4681/2023/11-ART36
https://doi.org/10.1145/3624480

**ACM Reference format:**
Francesco Spinnato, Riccardo Guidotti, Anna Monreale, Mirco Nanni, Dino Pedreschi, and Fosca Giannotti. 2023. Understanding Any Time Series Classifier with a Subsequence-based Explainer. *ACM Trans. Knowl. Discov. Data.* 18, 2, Article 36 (November 2023), 34 pages.
https://doi.org/10.1145/3624480

## 1 INTRODUCTION

The increasing availability of high-dimensional data stored in the form of time series such as electrocardiogram records, stock indices, motion sensors data, and so on, contributed to the diffusion of a wide range of time series classifiers [6, 80] in a variety of essential applications, ranging from the identification of stock market anomalies to the automated detection of heart diseases. The rising interest in this topic is confirmed by two surveys [6, 80] in which different kinds of univariate and multivariate time series classification models are tested and compared. The most common baseline, to which all other models are compared, is **k-Nearest Neighbors (KNN)** [59, 85, 97], usually paired with the Euclidean distance, **Dynamic Time Warping (DTW)** [72] or others [8, 98]. Transformation-based classifiers are also becoming very relevant, extracting different kinds of features from entire time series like BOSS [81] and WEASEL+MUSE [82], or sub-intervals like CIF [67], RISE [26] and FIT [60]. Further, the surveys focus both on traditional ensemble-based approaches like HIVE-COTE [63] and more recent deep learning-based models like ResNet [39] and TapNet [96]. Rocket [17], and its faster version MiniRocket [18], are regarded as the current best-performing state-of-the-art models. They use random convolutional kernels to quickly classify univariate and multivariate time series and have the best trade-off between speed and accuracy. The drawback of most of these models lies in their complexity, which makes them black-boxes and causes the non-interpretability of the internal decision process for humans [22]. However, when it comes to making high-stakes decisions, such as clinical diagnosis, the explanation aspect of the models used by **Artificial Intelligence (AI)** systems becomes a critical building block of a trustworthy interaction between the machine and human experts. Meaningful explanations [75] of time series classification would augment the cognitive ability of domain experts, such as medical doctors, to make informed and accurate decisions and better support AI accountability and responsibility in the decision-making.

A line of research exploring interpretable, transparent-by-design, and efficient time series classifiers is based on *shapelets* [95]. Shapelet decision trees [95] and shapelet transforms [62] extract shapelets from the time series of the training set by selecting subsequences with high discriminatory power and exploiting them for the classification process. Alternative approaches for mining discriminatory subsequences are the Matrix Profile [14] and SAX approximation [57, 83]. Unfortunately, in terms of accuracy and stability, all such methods lag far behind black-box time series classifiers, particularly in the presence of noisy data [25].

In this paper, we investigate the problem of *black-box explanation* for time series classifiers. We propose LASTS (**Local Agnostic subsequence-based Time series explainer)**, an **explainable AI (XAI)** method unveiling the logic of *any* black-box classifier operating on time series. Given a time series $X$ labeled with class $\hat{y}$ by a black-box $b$, LASTS returns an explanation $e$ composed of three parts that reveal the reasons for the opaque model's decision via different representations. First, a *saliency-based explanation* highlights the most important parts of the time series, which are responsible for driving the black-box towards the outcome $\hat{y}$ or away from it. Second, an *instance-based explanation* composed by a set of *exemplar* and *counterexemplar* time series. Exemplars are instances classified with the same label of $X$ and highlight common parts responsible for the classification. On the other hand, counterexemplars are instances similar to $X$ but with a different label and provide evidence of how the time series should be "morphed" for being classified with a

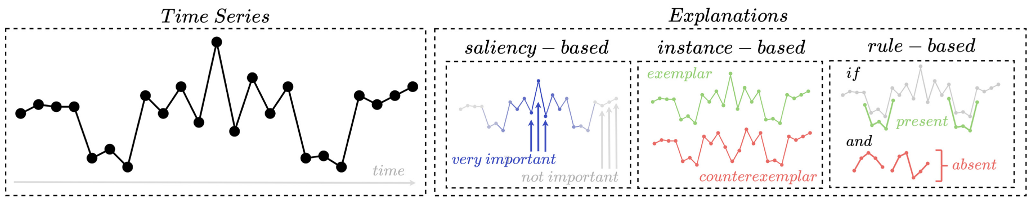

Fig. 1. An illustrative example of explanations obtained with LASTS. The black time series on the left is explained by *(i)* a saliency map, highlighting relevant and irrelevant points, *(ii)* an instance-based explanation, in which exemplars and counterexemplars can be compared, and *(iii)* by a rule-based explanation, showing interesting present and absent subsequences.

different label. Third, a *factual and counterfactual rule-based explanation* reveals the reasons for the classification through logic conditions expressed as *subsequences* that must (or must not) be contained in the time series in order to obtain (or not) the outcome $\hat{y}$. We emphasize the importance *counterfactual* components of the explanation of LASTS, i.e., counterexemplars and counterfactual rules, which are becoming essential ingredients in XAI methods [3, 12, 89]. While factual, direct explanations such as decision rules [53], and feature importance [64, 78], are crucial for understanding the reasons for a certain prediction, a counterfactual reveals what should change in a given instance to obtain a different classification outcome [89]. Counterfactuals are useful because they facilitate reasoning about the cause-effect relationships between observed features and classification outputs.

This work generalizes the approach proposed in [37] as a modular framework that is able to explain any black-box classifier for univariate and multivariate time series, also extending it in several ways.[1] In line with recent studies on XAI [64, 78], we tackle the time series black-box outcome explanation problem by deriving a *local* explanation to understand the behavior of the black-box in the *neighborhood* of the instance to explain [36]. Inspired by [34, 37], we develop a unified, modular framework and overcome many state-of-the-art limitations. First, we propose a novel neighborhood generation strategy (CFS) to generate consistent synthetic time series. For this purpose, we exploit autoencoders [40] for generating, encoding, and decoding a local neighborhood composed of exemplar and counterexemplar instances. Second, we present a novel way of generating a saliency map that does not require performing arbitrary time series segmentations, usually needed in competitor approaches [38, 64]. This saliency map provides a quick and immediate assessment of the crucial points of the time series for the classification, allowing a better understanding of the most critical observations. Third, we use the shapelet [62] and SAX [61] transformations paired with a local surrogate tree for designing meaningful rule-based explanations, useful and easy to understand [33], which are based on interpretable time series subsequences.

Thus, the explanation $e$ returned by LASTS has the unique characteristic of being simultaneously saliency-based, rule-based, and instance-based. Further, it explains the black-box decision by exploiting three different data granularities, i.e., time points, subsequences, and entire time series, contrary to other state-of-the-art XAI approaches, which are typically limited to one of the three [87]. The benefit of having heterogeneous explanations is that they can be adopted in various contexts to convey the reasoning behind a black-box decision to different types of users through multiple alternative forms. Figure 1 shows an illustrative example of the explanations provided by LASTS. First, the saliency-based explainer highlights the most significant or relevant points in a time series, providing a high-level understanding of key elements within the data. Such a tool

[1] This work extends "Explaining Any Time Series Classifier" presented at the IEEE International **Conference on Cognitive Machine Intelligence (CogMI)** 2020 [37].

could be useful for decision-makers or executives who need to understand key points of a time series prediction but do not have the time or need to investigate the finer details, such as project managers attempting to understand key progress markers or setbacks in a project timeline. Exemplars and counterexemplars are used in the instance-based explanation to present a complete data picture. They may be most useful for users seeking a comprehensive understanding of the data or experimenting with "what-if" scenarios: for example, strategists and planners interested in what happened and what could have happened in different circumstances. Finally, rule-based explainers provide explanations based on the presence or absence of specific subsequences in the data. Deciphering these patterns can have practical implications. For example, compliance officers tasked with detecting fraudulent activities in financial transactions must recognize specific patterns that indicate irregularities. Similarly, engineers monitoring production processes may look for operational anomalies that have specific shapes.

It is important to highlight the inherent heterogeneity of these explanations, each deriving from different methodologies and each offering unique insights into the time series data. This is both a strength and a challenge. On the one hand, it is a strength because each type of explanation complements the others by highlighting different aspects of the time series data. On the other hand, it poses a challenge as not all explanations are equally suited to represent all aspects of the data. For instance, saliency-based explainers are excellent at spotlighting the most significant points within a time series, providing a condensed understanding of key elements. However, their representation method, focused on what is present in the data, limits their ability to signify the importance of patterns not contained within the data. Conversely, due to their focus on the presence or absence of specific sequences, subsequence-based explainers can effectively highlight both existing patterns and the significance of absent ones, thus offering a different lens through which to view the data. Hence, a comprehensive understanding of the time series data necessitates amalgamating these heterogeneous explanations. By employing a diverse set of explainer types, the strengths of one can compensate for the limitations of others, ensuring a complete and nuanced understanding of the time series data. Each type of explanation, therefore, is not just a standalone analytical tool but an integral component of a comprehensive explanatory system. Overall, the effectiveness and relevance of each explainer type are contingent on multiple factors, such as the complexity of the time series data, the specificity of the questions posed, and the user's technical comprehension and individual needs. The primary advantage of using LASTS, as compared to separately adopting competitor explainers, is that all explanations, even though they are produced by different approaches, come from the same original source: the autoencoder.

To the best of our knowledge, LASTS is the only model-agnostic approach able to return a set of heterogeneous explanations, offering an in-depth understanding of the local decision of the black box. We present a wide experimentation, testing different alternatives to the proposed approach. We empirically demonstrate that LASTS provides faithful, stable, useful, and really understandable explanations by benchmarking it against state-of-the-art competitors on 15 time series datasets [64, 79].

To summarize, the main contributions of this work are:

- a local, model agnostic, framework for explaining time series classifiers: LASTS can explain the prediction of any univariate and multivariate time series classifier;
- a heterogeneous set of explanations for time series classification: a saliency map, subsequence-based factual and counterfactual rules, exemplar and counterexemplar time series instances;
- a quantitative evaluation on several datasets: LASTS is evaluated on univariate and multivariate datasets and against state-of-the-art competitors, demonstrating the effectiveness of each part of the explanations.

The rest of the paper is organized as follows. Section 2 discusses related works. Section 3 formalizes concepts used to design LASTS, which is then described in Section 4. Section 5 presents the experiments. Section 6 summarizes our contribution, its limitations, and future research directions.

## 2 RELATED WORK

Research on black-box explanation has recently received much attention [1, 36]. This growing interest is driven by the idea of incorporating opaque classifiers accompanied by explainers into AI systems, allowing the coexistence of high performance and explainability [68]. XAI approaches can be categorized according to many aspects [9]. First, a common taxonomy differentiates between *ante-hoc*, i.e., directly interpretable white-box models, and *post-hoc* explainability approaches, which explain black-box models after training without changing their underlying structure. XAI approaches can be further divided into *model-specific* if they exploit knowledge of the internal structure of the black-box and *model-agnostic* if they do not. Moreover, *local* XAI approaches provide explanations for a specific instance of the dataset, while *global* approaches explain the logic of the black-box as a whole. Finally, XAI approaches can be categorized depending on their explanation output. Many kinds of explanations exist, depending on the specific task, the problem domain, and most of all, the kind of data under analysis [9]. In our setting, i.e., time series classification, explanations can be divided into *time-step-based*, e.g., saliency maps, when they focus on the importance of each observation towards the classification output, *subsequence-based* when they explain the classification outcome using discriminative patterns of the time series, or *instance-based*, e.g., prototypes or counterfactuals, when they use whole time series to exemplify some salient property of the data.

XAI for univariate and multivariate time series data is a rising topic in the literature, with a plethora of approaches tackling the problem from different points of view. In the following, we focus on overviewing the most pertinent methods related to our work, i.e., post-hoc, model-agnostic, and local XAI approaches. On this point, one cannot fail to mention LIME and SHAP. LIME [78] randomly generates instances "around" the instance to explain, creating a local neighborhood. Then, it trains a linear model on the neighborhood labeled with the black-box. The explanation consists of the feature importance of the linear model. SHAP [64] connects game theory with local explanations and overcomes LIME's limitations, exploiting the Shapley values of a conditional expectation function of the black-box providing the unique additive importance for each feature. Methods like LIME and SHAP are thought for tabular data. However, if naively applied to time series classifiers, they can provide a time-point-based explanation for time series classifiers, considering each point as a separate feature [4]. Unfortunately, this procedure can work only on toy examples due to computation costs and given that time series classifiers are usually robust w.r.t. perturbations on single observations. Instead, a more involved approach is first to segment the time series and then use each segment as a feature [37, 38, 70]; however, the choice of the segmentation and type of perturbation is completely arbitrary. This paper proposes a saliency-based explanation that does not require any segmentation or discretization of the dataset analyzed.

Regarding subsequence-based explanations, the vast majority of XAI approaches are model-specific and usually extract patterns in the form of shapelets [95], or symbolic subsequences [61]. These subsequences with high discriminatory power are then used to transform the time-series dataset into a simpler representation [62] which can be paired with white-box classifiers such as decision trees or logistic regressors, guaranteeing an explanation for the decision. Given their simplicity, these approaches are generally lacking in terms of accuracy. There are many works in the literature aimed at improving the efficiency of the subsequence search using various optimization techniques [41, 42, 47], or by learning subsequences via gradient descent [30], or by first discretizing the time series using SAX [61] to speed up the search of greedy algorithms [56, 57].

Model-specific methods like ADSNs [65] and XCNN [91] take a different approach instead, directly generating patterns with adversarial learning to ensure their realism via a discriminator network. We reiterate that, differently from the aforementioned methods, our approach is model agnostic. We use subsequences to generate simple decision rules that explain the output of *any* black-box in terms of logical conditions. To the best of our knowledge, there are only a handful of XAI approaches that output rules as explanations, such as Anchor [79], and RuleMatrix [69], but none of them are tested on time series data.

Instance-based explanations are becoming more and more common in the literature. They exemplify a model's decision by providing salient and important instances that can directly explain the black-box decision, i.e., prototypes, or indicate the minimal changes that result in a different classification outcome, i.e., counterfactuals. Again, most of these approaches are model-specific, like [45] which can generate counterfactuals for k-nearest neighbor and Random Shapelet Forest [44] classifiers, or CEM [19], which uses an LSTM and a fully connected network to find the minimal perturbations that change the model's prediction. To the best of our knowledge, the only model-agnostic instance-based approaches for time series are *native guides* [16], LatentCF++ [92] and *CoMTE* [5]. *Native guides* extracts potential counterfactuals starting from the original dataset and adapts them to generate novel ones. LatentCF++ uses generative models to create counterfactual for Convolutional and LSTM networks. The main drawback of these approaches is that they only work on univariate time series data, contrary to our proposal. *CoMTE* can provide counterfactual explanations for multivariate time series classification by computing the minimal number of substitutions in order to change the predicted class of the original time series. Different from our approach, *CoMTE* can explain only black-boxes that return a prediction as class probability, thus precluding its applicability on widely used models like Rocket [17] and Minirocket [18] which are commonly combined with a Ridge classifier. Also, in our approach, the perturbations are performed in a so-called latent space to ensure the generation of a set of consistentcounterfactual instances.

## 3 SETTING THE STAGE

In this paper, we address the *black-box outcome explanation problem* [36] in the domain of time series classification. We keep our paper self-contained by summarizing the key concepts necessary to comprehend the proposed explanation method.

### 3.1 Time Series

A time series signal is defined as follows:

*Definition 3.1 (Time Series Signal).* A time series signal (or channel, dimension) $\mathbf{x}$ is a set of $m$ real-valued observations sampled at equal time intervals, $\mathbf{x} = \{x_1, \dots, x_m\} \in \mathbb{R}^m$.

A set of one or more time series signals forms a time series:

*Definition 3.2 (Time Series).* A time series $X$ is a set of $d$ signals, $X = \{\mathbf{x}_1, \dots, \mathbf{x}_d\} \in \mathbb{R}^{m \times d}$.

When $d = 1$, the time series is *univariate*, while if $d > 1$ the series is *multivariate*. A **Time Series Classification (TSC)** dataset is a set of time series with a vector of labels (or classes) attached.

*Definition 3.3 (TSC Dataset).* A *time series classification dataset* $\mathcal{D} = (\mathcal{X}, \mathbf{y})$ is a set of $n$ time series, $\mathcal{X} = \{X_1, \dots, X_n\} \in \mathbb{R}^{n \times m \times d}$, with a set of assigned labels, $\mathbf{y} = \{y_1, \dots, y_n\} \in \mathbb{N}^n$.

For a dataset $\mathcal{D}$ containing $c$ classes, $y_i$ can take $c$ different values. When $c = 2$, $\mathcal{D}$ is a binary classification dataset, while if $c > 2$, then $\mathcal{D}$ is a multi-class classification dataset. In order to ensure clarity and consistency in notation, we have adopted a tensor-like notation based on [52]. Lowercase letters are used to denote single scalar observations (e.g., $x$), while bold lowercase letters are used to denote vectors and individual signals of time series (e.g., $\mathbf{x}$). Capital letters are used to denote matrices and time series instances (e.g., $X$), and Euler script letters are used to denote tensors and time series datasets (e.g., $\mathcal{X}$). To indicate a specific observation in a time series dataset, we use the notation $x_{i,j,k}$, where $i$ denotes the $i^{th}$ multivariate time series in the dataset, $j$ denotes the $j^{th}$ time-step, and $k$ denotes the $k^{th}$ signal of the time series. Although time series do not always form a proper tensor since different channels can have different numbers of observations, we use a unique symbol $m$ to denote the length of the time series for simplicity of notation. When indexes are not relevant, we omit them for better readability. We can now define the TSC problem as:

*Definition 3.4 (TSC).* Given a TSC dataset $\mathcal{D}$, *Time Series Classification* is the task of training a function $f$ from the space of possible inputs to a probability distribution over the class values in $\mathbf{y}$.

The resulting TSC function $f$ takes as input a time series $X$ and returns $\hat{y}$ according to what $f$ learned, i.e., $\hat{y} = f(X)$. In general, $\hat{y}$ can either be a discrete label or the probability of $X$ belonging to a specific class. We use $f(\mathcal{X}) = \hat{\mathbf{y}}$ as a shorthand for $\{f(X) \mid X \in \mathcal{X}\} = \hat{\mathbf{y}}$. Typically, the classifier $f$ can be queried at will.

A common way to build classifiers in the time series domain is to use subsequences. In simple terms, a subsequence is a continuous sample of observations from a time series. Formally:

*Definition 3.5 (Subsequence).* Given a signal $\mathbf{x} = \{x_1, \dots, x_m\}$ of a time series $X$, a *subsequence* $\mathbf{s} = \{x_j, \dots, x_{j+l-1}\}$ of length $l$ is an ordered sequence of values such that $1 \leq j \leq m - l + 1$.

Subsequence-based classification approaches search for patterns that better discriminate the dataset labels, i.e., they try to find those subsequences that are most dissimilar between instances belonging to different classes. The two most common kinds of subsequences used for this purpose are shapelets [95] and symbolic subsequences [61].

***Shapelets.*** Typically, shapelet-based methods extract a set containing $p$-most discriminative shapelets by minimizing an information gain-like metric. Once the most discriminative shapelets are found, the Shapelet Transform [62] can be applied in order to transform the time series dataset into a simplified representation. Formally:

*Definition 3.6 (Shapelet Transform).* Given a time series dataset $\mathcal{X}$ and a set $S \in \mathbb{R}^{p \times l}$ containing $p$ shapelets, the *Shapelet Transform*, $\varsigma$, converts $\mathcal{X} \in \mathbb{R}^{n \times m \times d}$ into a real-valued matrix $T \in \mathbb{R}^{n \times p}$, obtained by taking the minimum distance between each time series in $\mathcal{X}$, and each shapelet in $S$, via a sliding-window, i.e., $T = \varsigma(\mathcal{X})$.

As a note, the sliding-window distance is method-dependent, and it is calculated w.r.t. the signal the shapelet was extracted from. In practice, the shapelet transform extracts the $p$-most discriminative shapelets from a time series dataset and returns a new representation of the data where the attributes represent the distances between each time series and the $p$ shapelets. Hence, any classification algorithm can be used, potentially increasing the accuracy while reducing training time.

***Symbolic Subsequences.*** Given that exhaustive subsequences search is computationally expensive, a common approach is to first transform the time series into a simplified representation by using SAX. The **Symbolic Aggregate approXimation (SAX)** algorithm [61] transforms time series into sequences of strings. For each time series signal $\mathbf{x} = \{x_1, \dots, x_m\} \in \mathbb{R}^m$, SAX uses the **Piecewise Aggregate Approximation (PAA)** [46] to split it into $w$ equally sized intervals and averages the values of each interval. Then, the time series signal is discretized using a finite set of symbols, i.e., an alphabet $\mathbb{A}$. Formally, $\text{SAX}(\mathbf{x}) = \tilde{\mathbf{x}} = \{\tilde{x}_1, \dots, \tilde{x}_w\} \in \mathbb{A}^w$, with $|\mathbb{A}| > 1$ being the number of symbols in the chosen alphabet. This approximation reduces running time while denoising the time series. Once the time series are converted to a symbolic representation, the most discriminative symbolic subsequences can be found using different techniques. Similar to the Shapelet Transform, the dataset can be transformed into a new representation, having as features the extracted subsequences and as values 0 or 1 depending on the absence or presence of subsequences inside the time series. Formally:

*Definition 3.7 (Symbolic Subsequence Transform).* Given a time series dataset $\mathcal{X}$ and a set $S \in \mathbb{A}^{p \times l}$ containing $p$ symbolic subsequences, the *Symbolic Subsequence Transform*, $\varsigma$, converts $\mathcal{X} \in \mathbb{R}^{n \times m \times d}$ into a binary-valued matrix $T \in \{0,1\}^{n \times p}$, obtained by checking if each subsequence in $S$ is contained or not in each time series in $\mathcal{X}$, i.e., $T = \varsigma(\mathcal{X})$.

For interpretability purposes, the symbolic subsequences can be easily mapped back to the original segments of the time series. We emphasize that we use the same symbol $\varsigma$ to denote the symbolic subsequence transform and the shapelet transform, given that they both convert a time series dataset into a tabular representation with subsequences as features.

## 3.2 Types of Explanations

Given a not interpretable, i.e., black-box, time series classifier $b$ and a time series $X$ classified by $b$, i.e., $b(X) = \hat{y}$, our aim is to provide an explanation $e$ for the decision $b(X) = \hat{y}$. More formally:

*Definition 3.8 (Time Series Black-box Outcome Explanation Problem).* Let $b$ be a not interpretable time series classifier, and $X$ a time series whose decision $\hat{y} = b(X)$ has to be explained, the *time series black-box outcome explanation problem* consists in finding an explanation $e \in E$ belonging to a human-interpretable domain $E$.

To build a complete, human-interpretable explanation in the time series domain, we consider three kinds of explanations: saliency maps, examples, and decision rules.

***Saliency Maps.*** Saliency maps are explanations that highlight the contribution of each feature for the classification [36]. Formally, for time series:

*Definition 3.9 (Saliency Map).* Given a time series $X$ a *saliency map* $\Phi = \{\phi_{1,1}, \dots \phi_{j,k}, \dots \phi_{m,d}\} \in \mathbb{R}^{m \times d}$ contains a score $\phi_{j,k}$ for every real-valued observation $x_{j,k}$ of $X$.

In practice, the saliency map assigns an importance score to each observation in $X$ depending on its contribution to the classification output.

***Examples.*** Examples, also called example-based (or instance-based) explanations, use whole time series objects to add interpretability to classification models by providing a comparison of the instance to explain with a salient time series. The two main different types of examples are *exemplar* and *counterexemplar* instances [34, 37]. Exemplars, also called prototypes, are time series that exemplify the main characteristics that influence the classifier's decision. Formally:

*Definition 3.10 (Exemplar).* Given a classifier $f$, an instance $X_=$ is an exemplar if there is a set of instances $\mathcal{X}' \subset \mathcal{X}$ *represented by* $X_=$, and such that $\forall X \in \mathcal{X}', f(X_=) = f(X)$.

The explanation is obtained by comparing the instance $X$ for which we have the decision $f(X)$ with the exemplar $X_=$ that represents it. Representation is usually formalized with a notion of similarity. On the other hand, counterexemplars, also called counterfactual instances, are very similar w.r.t. the instance to explain but are classified differently. The explanation is achieved by comparing the minimal differences in shape that lead to a different classification outcome. Formally:

*Definition 3.11 (Counterexemplar).* Given a classifier $f$ that outputs the decision $\hat{y} = f(X)$ for an instance $X$, a *counterexemplar* consists of an instance $X_{\neq}$ such that the decision for $f$ on $X_{\neq}$ is different from $\hat{y}$, i.e., $f(X_{\neq}) \neq \hat{y}$, and such that the difference between $X$ and $X_{\neq}$ is *minimal*, and that $X_{\neq}$ is *plausible*.

Minimality usually refers to a distance metric, while plausibility is assessed by checking if the instance is not simply an adversarial example [29] and is semantically coherent with the dataset.

***Decision Rules.*** Explainable AI methods increasingly simplify explanations to increase user trust by ensuring they identify cause-effect relations between events [12]. In this sense, rule-based explanations are arguably the most interpretable from a human standpoint as they allow the user to understand the reason behind a classifier's decision in terms of if-then statements. Formally:

*Definition 3.12 (Decision Rule).* Given an instance $X$, a decision rule is a function $r : p \rightarrow \hat{y}$, where the premise $p$ is a set of logical conditions on feature values, and $\hat{y} = r(X)$ is the predicted class value for $X$.

Decision rules are generated by rule-based classifiers or can be inferred by analyzing the splits of decision trees. They can be naively applied to time series by considering observations as features. In this case, a condition is in the form $(j,k) \in [v_{\text{low}}, v_{\text{up}}]$, where $(j,k)$ are the indexes identifying the $j^{th}$ time-step of the $k^{th}$ signal, and $v_{\text{low}}, v_{\text{up}} \in \bar{\mathbb{R}}$ are the lower and upper bounds on the observation value. A time series $X_i$ is covered by the rule if every condition in $p$ is true. A local rule-based model $rm$ can be used for explaining the black-box prediction for $X$ if it imitates sufficiently well its behavior, i.e., if $rm(X) = b(X)$ and also for every $X'$ in the neighborhood of $X$, $rm(X') = b(X')$. The definition of the neighborhood is method-dependent. A rule that directly explains the prediction of a black-box is called a factual rule. In contrast, the rules obtained by minimally removing or adding conditions in the factual rule premises are called counterfactual rules. Counterfactual rules are extremely useful for what-if analysis because they allow understanding of the minimal variation that results in a different classification by the black-box. The most notable rule-based XAI approaches for tabular data are Anchor [79], a model-agnostic method explaining the behavior black-boxes with high precision factual rules, and LORE [35], which generates the local neighborhood via a genetic algorithm, trains a decision tree to then extract factual and counterfactual decision rules. The first version of LASTS [37] extends LORE to univariate time series data, generating rules that explain the decision of the black-box in terms of logical conditions based on time series subsequences. In this case, the conditions in the premise refer to subsequences instead of single time series observations, and the feature values represent the presence/absence of subsequences inside the time series. We build upon this to extend the approach to multivariate time series data.

## 3.3 Autoencoders in XAI

A standard **autoencoder (AE)** [40] is a type of neural network trained for learning a representation that reduces the dimensionality from $m \times d$ to $k$ and captures non-linear relationships. An *encoder* $g : \mathbb{R}^{m \times d} \rightarrow \mathbb{R}^q$, and a decoder $h : \mathbb{R}^q \rightarrow \mathbb{R}^{m \times d}$ are simultaneously trained with the objective of minimizing the *reconstruction loss*. Starting from the encoding $\mathbf{z} = g(X)$, the autoencoder tries to reconstruct a representation as close as possible to its original input $X \simeq h(\mathbf{z}) = \hat{X}$. Autoencoders learn to encode their input in a latent representation, which is usually of smaller dimensionality and, therefore, simpler and easier to deal with, w.r.t. the original input. The latent space can also be used to sample synthetic instances that can be decoded into completely new and unseen records. In order to use autoencoders as generators, it is useful to have an encoder with a specific latent distribution from which to sample. In this sense, the two leading solutions used in the literature are **Variational Autoencoders (VAEs)** [51], which learn the parameters of a latent distribution, usually the mean and standard deviation of a Gaussian, or **Adversarial Autoencoders (AAEs)** [66], that use a GAN inspired approach [28], adding a discriminator in the network architecture, trained to discriminate the latent distribution, constraining it to the desired form.

In order to support the generation of a good explanation, the autoencoder used in our proposal needs to have some desirable properties:

(1) a known latent distribution to sample consistentsynthetic instances;
(2) a latent distribution that allows for the generation of meaningful neighborhoods around sampled points [2];
(3) a latent space that has as few dimensions as possible, to facilitate the neighborhood generation;
(4) good performances in the *(i)* reconstruction error and *(ii)* reconstruction accuracy, i.e., *(i)* the reconstructed instances need to be similar to the original instances, and *(ii)* the autoencoder must be good enough for the black-box to be able to predict the same class before and after the autoencoding.

The first property can be ensured by using VAEs or AAEs. These are superior to traditional AEs as generative models because they allow sampling from a known latent distribution. Usually, a VAE is easier to train w.r.t. an AAE, given the former has to optimize only one loss function, while the latter also has to consider the discriminator [34, 66]. For this reason, we adopt VAEs by default as autoencoders for the proposed approach. The second property can be ensured by checking the sampled instances with a discriminator or by using specific sampling techniques that minimize the distribution mismatch [2] (Section 3.4). The third and fourth properties can be achieved with hyperparameter tuning. In our setting, the reconstruction error is measured in terms of mean squared error between the original instances and the reconstructed ones, $\text{rec}_{\text{mse}} = \frac{1}{n}\sum_{i=1}^{n}(X_i - \hat{X}_i)^2$. The reconstruction accuracy is the percentage of instances in $\mathcal{X}$ that are correctly classified by the black-box after being reconstructed by the autoencoder. Formally, $\text{rec}_{\text{acc}} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{1}_{b(X_i)=b(\hat{X}_i)}$, where $\mathbb{1}$ is a function which outputs 1 if the condition in the subscript is true and 0 otherwise.

### 3.4 Neighborhood Generation in XAI

Neighborhood generation, being a central issue in the definition of local surrogates, is an increasingly studied topic in the domain of XAI [37, 64, 78]. Local surrogates mimic the local decision boundary of a black-box for the single instance they are tasked to explain. In order to be able to imitate the black-box, a representative neighborhood of the instance to explain has to be defined. It should be composed of *(i)* similar, i.e., spatially close, instances having the same label as the instance to explain (prototypes, exemplars), but it also should contain *(ii)* close instances having a different label (counterfactual instances, counterexemplars, distractors). Post-hoc interpretability approaches often rely on perturbations of the input data that query the black-box to understand how its prediction changes. Two of the most notable examples are LIME [78] and SHAP [64]; however, the perturbation methods used by these models do not ensure the generation of realistic data. Moreover, even if these approaches can be applied to time series data with some modifications, they are not explicitly thought for it [34, 37, 54].

For time series data, it is often hard to generate a meaningful neighborhood in the manifest space by directly perturbing the original instances because the risk of generating unrealistic adversarial examples is relatively high. This is the main reason why generative models such as VAEs and AAEs are increasingly used in this field [7, 34, 37, 43, 54, 74]. In fact, these models are usually trained to produce a specific prior distribution in the latent space, typically a Gaussian. After the training phase, they can be used as generators by sampling new instances or perturbing existing ones. Perturbation techniques can greatly vary: from the usage of different sampling or searching algorithms [74], to genetic approaches [34, 37], to gradient descent-based methods [7, 43]. Even if the technical aspects of these methods are very different, they all rely on autoencoders to generate a suitable latent space to explore and analyze.

In [93], it is shown that latent space operations commonly used in the literature, such as instance interpolation and vicinity sampling, induce a so-called *distribution mismatch* between the outputs and the prior distribution the model was trained on. This is a delicate issue, given that decoders and generators are usually trained on fixed priors and thus assume that their inputs will have statistical properties that align with their distributions. For this reason, in [2] is proposed a Gaussian-matched neighborhood generation function which avoids the sampling from latent space locations that are highly unlikely given the prior distribution of the autoencoder, and allows the creation of more consistent, i.e., higher quality synthetic instances. Given a latent space vector $\mathbf{z} \in \mathbb{R}^q$ in which each scalar entry is independently sampled from a standard Gaussian distribution, i.e., $\forall z \in \mathbf{z},\ z \sim \mathcal{N}(0,1)$, a standard **Gaussian (GA)** vicinity sampling is defined as $\mathbf{z}_{\text{GA}} = \mathbf{z} + \theta\mathbf{u}$ with $\theta$ being a scaling factor and $\mathbf{u} \in \mathbb{R}^q$ a randomly sampled normal Gaussian vector. The **Gaussian-matched (GM)** operation is defined as $\mathbf{z}_{\text{GM}} = \mathbf{z}_{\text{GA}}/\sqrt{1+\theta^2}$. In [2] it is shown that this approach is guaranteed to produce samples coming from a standard Gaussian distribution, i.e., samples that are indistinguishable from randomly sampled instances from that distribution. The authors show that this property is independent of the number of latent dimensions, and it is proven to work even in high-dimensional latent spaces. Incorporating this operation into our framework helps to reduce the distribution mismatch and generate a synthetic neighborhood that is more consistent, resulting in samples that seem as if they were drawn from the original training set.

## 4 LOCAL AGNOSTIC TIME SERIES EXPLAINER

In this section we present LASTS, a Local Agnostic subsequence-based Time series explainer, solving the black-box outcome explanation problem. Given a black-box $b$ and a univariate or multivariate time series $X$, the human-interpretable explanation $e \in E$ returned by LASTS for the classification $\hat{y} = b(X)$ is composed by three parts: *(i)* a saliency map highlighting the most sensible part of the time series, *(ii)* a set of *exemplars* and *counterexemplars*, and *(iii)* subsequence-based *factual* and *counterfactual* rules. The saliency map highlights the observations that are most responsible for a class change. Exemplars and counterexemplars illustrate time series classified with the same and with a different outcome than $X$. They can be visually analyzed to understand the reasons for the classification and make comparisons between $X$ and them. Finally, the factual rule shows the subsequences contained (and not contained) in $X$ responsible for the class $\hat{y}$, and vice-versa, the counterfactual highlights how the rule should change to have a different classification outcome. The explanation returned by LASTS satisfies the requirements of counterfactuability, usability, and meaningfulness [12, 68, 75], and offers to the final user a multi-modal explanation unveiling the reasons for the classification in different and complementary ways. A simple schema of LASTS can be viewed in Figure 2.

Besides the black-box $b$ and the time series $X$, LASTS requires a trained encoder $g$ and decoder $h$ for modeling times series in a simplified representation. The explanation process of LASTS, described in Algorithm 1 and in Figure 2, involves the following steps. First, LASTS encodes the time series $X$ in its latent representation $\mathbf{z}$ (line 1). Then, through the CFS function detailed in Algorithm 2, it searches for the closest instance to $\mathbf{z}$ having a different class, i.e., the closest counterexemplar $\mathbf{z}_{\neq}$ (line 2). Once $\mathbf{z}_{\neq}$ is found, LASTS generates a synthetic neighborhood $Z$ around it using the *neighgen* function, exploiting the distribution of the VAE (line 3). After that, the latent instances are decoded and labeled using the black-box, i.e., $\hat{\mathbf{y}} = b(\hat{\mathcal{X}})$ (line 4). The decoded neighborhood is used to get exemplar and counterexemplar time series (line 6), while the closest counterexemplar is used to compute the saliency map (line 5). The neighborhood is then represented as the

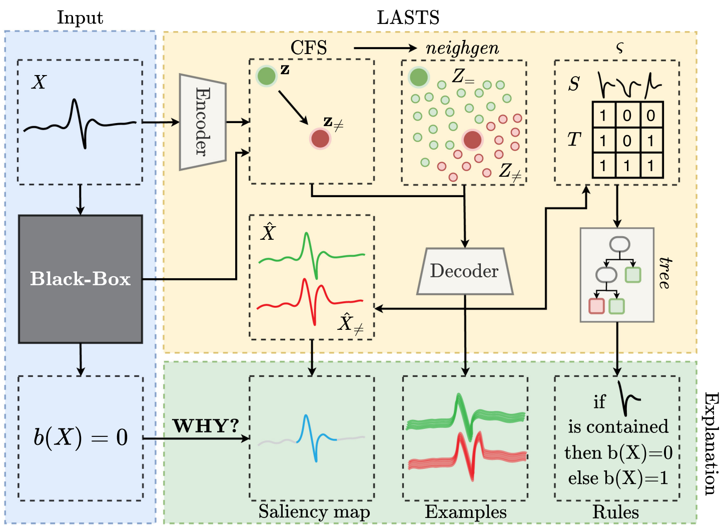

Fig. 2. A schema of LASTS. (*left*) the input of LASTS is composed of an instance $X$ and the black-box prediction for that instance, $b(X)$. (*bottom-right*) the output of LASTS is an explanation $e$ composed of a saliency map, exemplars and counterexemplars, and factual and counterfactual decision rules. (*top-right*) LASTS exploits the latent encoding of a VAE to perturb the time series and generate a synthetic neighborhood around the local decision boundary. Once decoded and classified by the black-box, these synthetic instances represent meaningful exemplar and counterexemplar time series and are used to construct a saliency map. Finally, interpretable subsequences are extracted from this neighborhood and used to train a decision tree surrogate, from which the factual and counterfactual rules are inferred.

**ALGORITHM 1:** LASTS$(X, b, g, h)$

```
Input  : $X$ - time series, $b$ - black-box, $g$ - encoder, $h$ - decoder,
Output : $e$ - explanation
1  $\mathbf{z} \leftarrow g(X)$;                                              // encode X into the latent space
2  $\mathbf{z}_{\neq} \leftarrow$ CFS$(\mathbf{z}, b, h)$                      // find closest counterfactual instance to z
3  $Z \leftarrow neighgen(\mathbf{z}, \mathbf{z}_{\neq})$;                     // generate latent neighborhood
4  $\hat{\mathcal{X}}, \hat{X}, \hat{X}_{\neq} \leftarrow h(Z), h(\mathbf{z}), h(\mathbf{z}_{\neq})$; $\hat{\mathbf{y}} \leftarrow b(\hat{\mathcal{X}})$;   // decode and classify neighborhood
5  $\Phi \leftarrow extractSaliency(\hat{X}, \hat{X}_{\neq})$                  // extract saliency map
6  $\hat{\mathcal{X}}_{=}, \hat{\mathcal{X}}_{\neq} \leftarrow extractExamples(\hat{\mathcal{X}}, \hat{\mathbf{y}})$   // extract (counter)exemplar instances
7  $T \leftarrow \varsigma^*(\hat{\mathcal{X}}, \hat{\mathbf{y}})$;            // subsequence transform
8  $dt \leftarrow tree(T, \hat{\mathbf{y}})$;                                 // learn decision tree surrogate
9  $r_{=}, r_{\neq} \leftarrow extractRules(\hat{X}, dt)$;                    // extract subsequence-based rules
10 return $e = \{\Phi, (\hat{\mathcal{X}}_{=}, \hat{\mathcal{X}}_{\neq}), (r_{=}, r_{\neq})\}$;   // return explanation
```

presence/absence of subsequences in a time series through the function $\varsigma^*$, which extracts the most discriminative subsequences and performs the subsequence transform $\varsigma$, obtaining the set $T$ (line 7). Finally, a decision tree $dt$ is trained on $(T, \hat{\mathbf{y}})$ (line 8) and used to retrieve the subsequence-based factual rule and counterfactual rules $r_{=}, r_{\neq}$ (line 9). The explanation comprises the saliency map, exemplar and counterexemplar instances, and the factual and counterfactual rules (line 10). Details of each step are presented in the rest of this section.

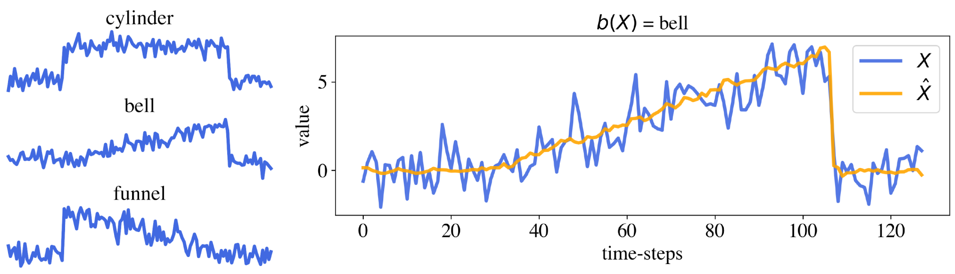

Fig. 3. Running example. (*left*) cylinder-bell-funnel instances. (*right*) time series $X$ to explain and its reconstructed version $\hat{X}$.

**ALGORITHM 2:** CFS($\mathbf{z}, b, h, n_s$)

```
    Input  : $\mathbf{z}$ - encoded version of $X$, $b$ - black-box, $h$ - decoder, $n_s$ - instances to search at each iteration
    Output : $\mathbf{z}_{\neq}$ - closest counterfactual to $\mathbf{z}$
1 $\theta \leftarrow \theta_0$;                                              // init. threshold
2 $Z \leftarrow sample(\mathbf{z}, n_s, \theta)$;                              // latent sampling around z
3 $Z_{\neq} \leftarrow []$;                                                    // init. counterfactual set
4 while $\exists\, \mathbf{z}' \in Z \mid b(h(\mathbf{z}')) \neq b(h(\mathbf{z}))$ do
5     $\theta \leftarrow \theta/2$;                                            // decrease threshold
6     $Z \leftarrow sample(\mathbf{z}, n_s, \theta)$;                          // latent sampling around z
7     $Z_{\neq}.add(\{\mathbf{z}' \in Z \mid b(h(\mathbf{z}')) \neq b(h(\mathbf{z}))\})$;   // store counterfactual instances
8 $\mathbf{z}_{\neq} \leftarrow closest(\mathbf{z}, Z_{\neq})$;                // get closest counterfactual instance to z
9 return $\mathbf{z}_{\neq}$;
```

## 4.1 Latent Encoding

The time series $X$ is passed to the encoder part of the VAE that compresses it into a so-called latent representation $\mathbf{z} = g(X)$. The time series in Figure 3 is used as a running example. In our case, $X$ originally has 128 observations, and it is compressed in a bidimensional vector, $\mathbf{z} = [1.344, -2.005]$, depicted in Figure 4 (left). The latent vector $\mathbf{z}$ can also be passed to the decoder to reconstruct the original time series. Both $X$ and its reconstructed version, $\hat{X} = h(\mathbf{z})$, are depicted in Figure 3 (right). The original time series is much more noisy w.r.t. its reconstructed version. This suggests that the autoencoder is able to capture the most relevant features of the time series for the classification, i.e., its general shape, discarding the random noise.

## 4.2 Counterfactual Search

The second step of LASTS is to find the closest counterfactual instance w.r.t. $\mathbf{z}$. To perform this search, LASTS adopts an algorithm that iteratively samples latent instances around $\mathbf{z}$, decodes them, and checks their label using $b$. The **counterfactual search algorithm (CFS)** is reported in Algorithm 2. CFS uses a function *sample* to sample a distribution of synthetic instances around $\mathbf{z}$. The function *sample* can theoretically be any sampling function, ranging from a pure random approach like in LIME [78] to a genetic algorithm maximizing a fitness function like in LORE [33]. A good generation function is fundamental to create consistentsynthetic instances by avoiding the aforementioned distribution mismatch [2], i.e., the sampling from locations in the latent space that are highly unlikely given the prior distribution of the autoencoder. We emphasize that we employ the term "consistent" as mentioned in [2], i.e., as a synonym for probable, likely, or, in other words, coherent with the autoencoder's underlying distribution. Our aim is to convey that the generated

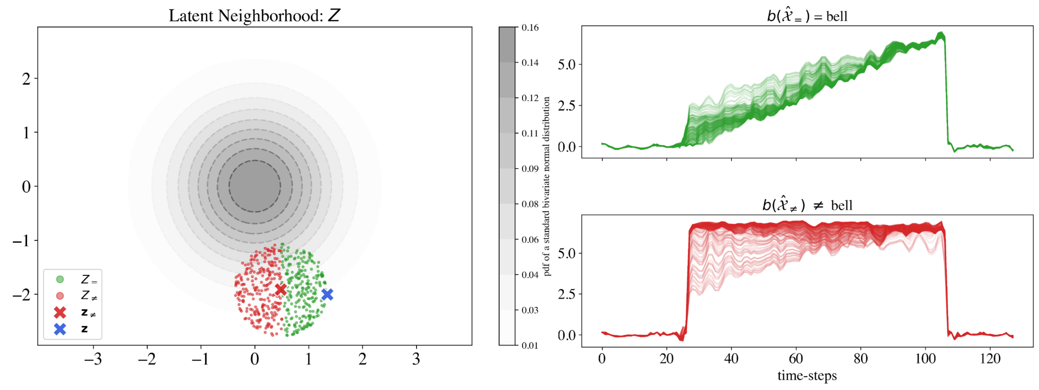

Fig. 4. (*left*) generated latent neighborhood compared to a standard normal bivariate distribution. (*right-top*) exemplar and (*right-bottom*) counterexemplar instances.

samples possess a similar distribution to that of the original training data. The first version of LASTS presented in [37] uses the *genetic* approach of LORE; in the version presented in this paper, LASTS adopts by default a procedure inspired by the *growing sphere* algorithm proposed in [55].

CFS depends on a threshold $\theta$ that decides the amount of space around $\mathbf{z}$ that the generated neighborhood is going to occupy. This threshold can be the *radius* for a spherical random uniform, as in the original growing sphere [55], or a scaling factor for a Gaussian distribution, as detailed in Section 3.4. The goal of CFS is to systematically explore the latent space in order to find the instance closest to $\mathbf{z}$ belonging to a different class, i.e., the closest counterfactual. The counterfactual search is performed by iteratively generating a neighborhood $Z$ around $\mathbf{z}$, using the aforementioned *sample* function, and by checking if there is at least one counterfactual among the generated instances. The presence of counterfactuals indicates that the sampled neighborhood is still crossing the decision boundary; therefore, the threshold is halved, and the procedure loops until the sampling function does not generate any counterfactual. We highlight that in this setting, the generation happens in the latent space, while the presence of counterfactuals is checked in the time series domain by decompressing the latent instances through $h$. At each step of the iteration, the CFS algorithm stores all counterfactuals generated and, once out of the loop, it selects the closest one to $\mathbf{z}$, i.e., $\mathbf{z}_{\neq}$, as the best counterfactual. In the running example, the closest counterfactual instance can be viewed in its latent form in Figure 4 (*left*).

## 4.3 Neighborhood Generation

The third step we detail is the generation of the neighborhood around $\mathbf{z}_{\neq}$ using the function *neighgen*. In principle, *neighgen* can be a different sampling function w.r.t. the *sample* function used for the search. In Section 5, we experiment with different combinations of the two. The sampling is performed extremely close to the black-box decision boundary; therefore, this synthetic neighborhood contains, by construction, both a set of time series having the same class as $\mathbf{z}$ and a set of time series having a different class, i.e., exemplars, $Z_=$, and counterexemplars, $Z_{\neq}$ (Figure 4, left). The number of distinct counterexemplar classes is influenced by the final threshold, and increases as $\theta$ increases. Note that the sampling is performed in the latent space, while the labels are retrieved by decoding the latent neighborhood and then applying the black-box function to obtain its predictions. For ease of viewing, we depict all the counterexemplar instances with the red color, independently of their specific class. $Z$ is decoded and classified by the black-box into $\hat{X}_=$ and $\hat{X}_{\neq}$. These instances represent the example-based explanation, showing how the decision

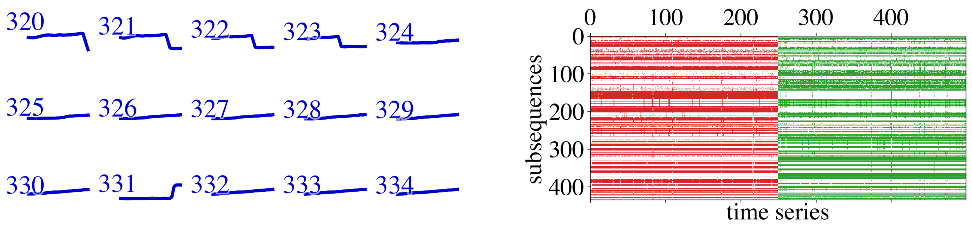

Fig. 5. (*left*) examples of subsequences with identifiers. (*right*) subsequence-transformed dataset $T$. Each column represents a time series. For each row, the cell is white for subsequences that are not contained, green for exemplars, and red for counterexemplars.

of the black-box changes depending on the shape of the time series (Figure 4, right). Moreover, by computing the absolute difference between the decoded closest counterfactual $\hat{X}_{\neq} = b(\mathbf{z}_{\neq})$, and $\hat{X} = b(\mathbf{z})$, we can discover the time points for which a change in the values is more likely to modify the decision of the black-box. Thus, the *saliency map* is defined as $\Phi = |\hat{X} - \hat{X}_{\neq}|$.

Figure 4 *(left)* shows the latent neighborhood $Z$ highlighting in green the exemplar instances labeled as *bell*, and in red the counterexemplar instances labeled with a different class value, in this case *cylinder*. In gray is depicted a standard bivariate normal distribution, highlighting the probability of hidden vectors in the latent space; formally $\mathcal{N}_2(\boldsymbol{\mu}, \Sigma)$ with $\boldsymbol{\mu} = [0, 0]^T$ and $\Sigma = I_2$ where $I_2$ is a 2-dimensional identity matrix. For ease of viewing, we provide an explanation for an instance at the edge of the distribution. The corresponding instances in the manifest space can be viewed in the same figure to the right. The synthetic neighborhood sampled in the latent space perfectly summarizes the separation of the different class values, unveiling a local decision boundary that is easy to detect even with a simple classifier. Both $Z_{=}$ and $Z_{\neq}$, forming the neighborhood, densely surround $\mathbf{z}_{\neq}$, helping to capture the black-box behavior locally around the closest decision boundary to $\mathbf{z}$. Note that $\mathbf{z}$ remains at the edge of the generated neighborhood by design because our area of interest is the decision boundary and not the instance to explain itself.

## 4.4 Subsequence Extraction

Given the decoded local neighborhood $\hat{\mathcal{X}}$ and $\hat{\mathbf{y}} = b(\hat{\mathcal{X}})$, LASTS extracts a set of subsequences $S$ and performs the subsequence transform $\varsigma^*$, encoding time series into a space of presence/absence of subsequences (line 7, Algorithm 1), as detailed in Section 3.1. Subsequences can be of many kinds; in Section 5, we test both shapelet-based and SAX-based subsequences. Figure 5 presents an example of SAX-based subsequences and a heatmap depiction of the transformed dataset. Each column represents a time series in $\hat{\mathcal{X}}$. For each row, a colored cell indicates the presence of the corresponding subsequence, while a white cell indicates its absence. The colored cells, green and red, indicate the SAX subsequences contained by the exemplar and counterexemplar time series, respectively. In some cases, we can observe a sort of complementarity. In other cases, a time series can simultaneously contain a combination of subsequences describing exemplars and counterexemplars.

## 4.5 Local Surrogate

Given $T$ and $\hat{\mathbf{y}}$, LASTS trains a subsequence-based decision tree classifier $dt$ that allows to identify subsequence-based factual and counterfactual rules $r_{=}, r_{\neq}$ (lines 8 - 9, Algorithm 1). LASTS adopts decision trees because factual and counterfactual decision rules can be naturally derived by following the root-leaf paths [33]. Figure 6 (*bottom*) reports the explanation rules for our example: $r_{=} = \{s_{327} \in X\} \rightarrow bell$, $r_{\neq} = \{s_{327} \notin X\} \rightarrow \neg bell$. The visual representation of the rules shows the position of the subsequences that must be contained and those that must not be contained at

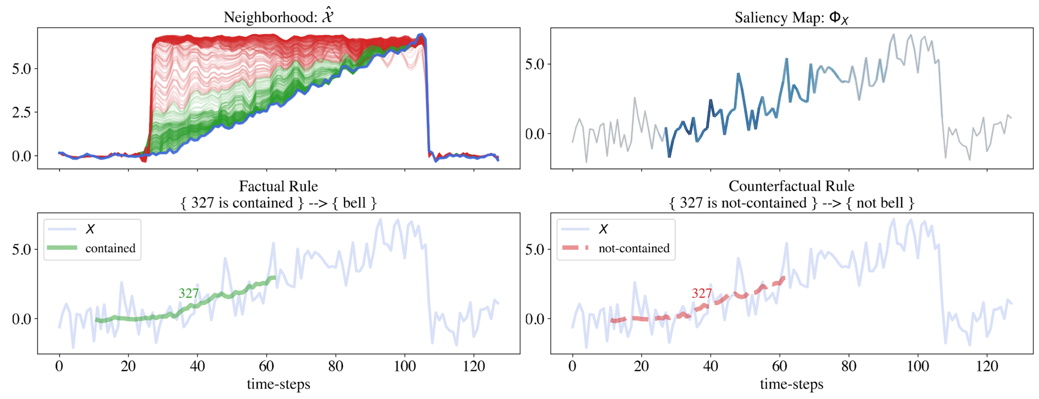

Fig. 6. (*top-left*) synthetic neighborhood with exemplar and counterexemplar instances. (*top-right*) saliency map highlighting the most sensible points for the classification. (*bottom*) factual counterfactual rule.

their best alignment with $X$. Looking at the rules, a user can truly understand the reasons for the classification. In this case, the presence of an increasing pattern in the time series differentiates between a *bell* instance and instances of other classes. In general, rules can be longer and include an arbitrary number of contained and non-contained subsequences, depending on the complexity of the classification task and the resulting surrogate tree.

## 4.6 Explanation

In summary, LASTS explains the prediction of a black-box $b$ for a time series instance $X$. The final explanation, $e = \{\Phi, (\hat{\mathcal{X}}_=, \hat{\mathcal{X}}_{\neq}), (r_=, r_{\neq})\}$, is composed of three parts. First, a saliency map, $\Phi$, highlights the most important timesteps of the time series, i.e., those timesteps that, if changed, would bring the prediction toward a different class. Second, a neighborhood composed of exemplar and counterexemplar instances. Exemplars, $\hat{\mathcal{X}}_=$, are instances close to $X$ in the latent space, with the same label as $X$. They are prototypes, showing the main characteristics of a specific class. On the other hand, counterexemplars, $\hat{\mathcal{X}}_{\neq}$, are instances close to $X$ in the latent space but classified by the black-box differently and can help the user understand how the prediction of the black-box changes by changing the shape of the time series consistently. Finally, the factual, $r_=$, and counterfactual, $r_{\neq}$, rules logically explain the prediction of the black-box, both in direct and contrastive ways, showing the minimum change in contained/not contained subsequences to modify the black-box prediction. The complete explanation for our running example is presented in Figure 6.

## 5 EXPERIMENTS

We experiment with LASTS both quantitatively and qualitatively. First, in Section 5.3, we compare different alternatives for neighborhood generation and subsequence extraction on 4 univariate datasets and 2 black-box models. Once the best framework combination is found, we benchmark it on 15 datasets, 10 univariate and 5 multivariate, respectively, from the UCR and UEA time series machine learning repositories[2]. Each part of the explanation returned by LASTS is evaluated. The instance-based part of the explanation is assessed through *usefulness* (Section 5.4). Then, the saliency-based part of the explanation is tested w.r.t. *stability*, *correctness* and by running *insertion/deletion* benchmarks against SHAP (Section 5.5). Furthermore, the rule-based part of the

[2]https://www.timeseriesclassification.com/

Table 1. Dataset Details, Autoencoder Performance in Terms of Reconstruction Accuracy and MSE, Rocket Performance in Terms of Accuracy

| DATASETS | | | DETAILS | | | | | | AUTOENCODER | | | ROCKET |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *name* | *abbr.* | *ref.* | $n_{\text{train}}$ | $n_{\text{test}}$ | $n_{\text{exp}}$ | $m$ | $d$ | $c$ | $q$ | $rec_{\text{mse}}$ | $rec_{\text{acc}}$ | *acc* |
| ArticularyWordRecognition | ART | [90] | 275 | 300 | 50 | 144 | 9 | 25 | 32 | 0.492 | 0.95 | 0.99 |
| Cylinder-Bell-Funnel | CBF | [23] | 268 | 84 | 36 | 128 | 1 | 3 | 2 | 1.067 | 1 | 1 |
| Cylinder-Bell-Funnel-Multi | CBM | [23] | 268 | 84 | 36 | 128 | 3 | 3 | 2 | 1.029 | 1 | 1 |
| Coffee | COF | [11] | 28 | 28 | 28 | 286 | 1 | 2 | 4 | 0.004 | 1 | 1 |
| ECG200 | EC2 | [73] | 100 | 100 | 50 | 96 | 1 | 2 | 4 | 0.180 | 0.99 | 0.90 |
| ECG5000 | EC5 | [27] | 500 | 4500 | 50 | 140 | 1 | 5 | 2 | 0.138 | 0.98 | 0.95 |
| ERing | ERI | [94] | 30 | 270 | 50 | 65 | 4 | 6 | 16 | 0.513 | 0.98 | 0.99 |
| GunPoint | GUN | [77] | 50 | 150 | 50 | 150 | 1 | 2 | 4 | 0.054 | 1 | 1 |
| ItalyPowerDemand | ITA | [48] | 67 | 1029 | 50 | 24 | 1 | 2 | 2 | 0.066 | 0.96 | 0.97 |
| Libras | LIB | [20] | 180 | 180 | 50 | 45 | 2 | 15 | 16 | 0.002 | 0.91 | 0.91 |
| PenDigits | PEN | [21] | 7494 | 3498 | 50 | 8 | 2 | 10 | 4 | 59.194 | 0.95 | 0.98 |
| PhalangesOutlinesCorrect | PHA | [15] | 1800 | 858 | 50 | 80 | 1 | 2 | 16 | 0.002 | 0.98 | 0.84 |
| Plane | PLA | [86] | 105 | 105 | 50 | 144 | 1 | 7 | 2 | 0.033 | 1 | 1 |
| Strawberry | STR | [13] | 613 | 370 | 50 | 235 | 1 | 2 | 4 | 0.002 | 0.99 | 0.98 |
| TwoLeadECG | TWO | [27] | 23 | 1139 | 50 | 82 | 1 | 2 | 2 | 0.042 | 1.00 | 1.00 |

explanation is evaluated using *fidelity*, *precision* and *coverage* against a global SAX-based decision tree surrogate (GLO-SAX) and against ANCHOR [79] re-adapted for time series (Section 5.6). Finally, in Section 5.7, we propose two qualitative examples of the explanation of LASTS on one univariate and one multivariate time series.

## 5.1 Datasets and Black-box Models

In Table 1, all the information about the datasets used for evaluating our approach is reported. The training set $\mathcal{X}_{\text{train}}$ is used both to train the black-box and the autoencoders. In principle, the dataset used for the black-box training, $\mathcal{X}_{\text{bb}}$, and the dataset used for the autoencoder training, $\mathcal{X}_{\text{ae}}$, can be different. However, due to the small dimensionality of some datasets, we set $\mathcal{X}_{\text{bb}} = \mathcal{X}_{\text{ae}} = \mathcal{X}_{\text{train}}$ for all the datasets, with the exception CBF and CBM which are synthetic, and their instances can be sampled at will. The test set $\mathcal{X}_{\text{test}}$ is used to benchmark both autoencoders and black-box models, and to sample $\mathcal{X}_{\text{exp}}$, which is composed of a maximum of 50 instances to explain and evaluate.

To test different framework alternatives, we train and explain a *ResNet* [39] (RES) implemented in `keras` according to [24], and a *k-Nearest Neighbor* [85] (KNN) baseline as implemented by `scikit-learn`. Once the best framework setup is found, we choose as a black-box to explain Rocket [17], as implemented by `sktime`, using the default parameters for the transform and a RidgeClassifierCV as classification model. The ResNet comprises three residual blocks, each containing three convolutional layers, a global average pooling layer, and a dense layer. The layers inside each residual block have respectively 64, 128, 256 filters, of size 8, 5, 3. We train RES with a batch size of 16, monitoring the loss, with a patience parameter of 50 epochs. As optimizer, we select Adam, with the default `keras` parameters: $learning\_rate = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, minimizing the sparse categorical crossentropy. For KNN we use the Euclidean distance and the $k$ parameter is selected via grid-search with $k \in [1, |\mathcal{X}_{\text{bb}}|]$.

## 5.2 Implementation Details

In the following, we specify the implementation details adopted for each module of LASTS[3].

[3]Code available at github.com/fspinna/lasts

***Variational Autoencoder.*** As autoencoder, we adopt a VAE for the reasons discussed in Section 3. For simplicity, we use the same network structure for all the datasets, i.e., a convolutional autoencoder composed of 16 layers, 8 for the encoder, and 8 for the decoder. The number of filters per layer is 8, and the kernel size is set to 3. The latent dimension is chosen on a dataset basis by starting with $q = 2$ and iteratively building autoencoders with an increasing value of $q$ until the accuracy for the reconstructed instances increases and reaches 0.90, but also keeping $q \leq m/2$. As an activation function, we always use ReLU. As optimizer, we select Adam with default `keras` parameters: *learning_rate* $= 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, minimizing the **Mean Squared Error (MSE)** with the Kullback–Leibler divergence regularization term. The VAEs are trained with a batch size equal to 32, for a maximum of $8,000$ epochs, monitoring the validation loss with a patience of 500 epochs before halving the learning rate and a patience of $1,250$ epochs before early stopping. We measure the performance of the autoencoders through the reconstruction error between the original and reconstructed time series, in terms of *Mean Squared Error* (the lower, the better), and in terms of *accuracy* of the classifiers on the reconstructed time series (the higher, the better). Since the encoding operation in a VAE is stochastic, $\mathbf{z}$ can vary slightly. Therefore, to improve the stability of the framework, $X$ is encoded 1000 times, and the latent representation $\mathbf{z}$ is chosen by checking the most similar reconstruction $h(\mathbf{z})$ to $X$ using the Euclidean distance. Given a set of $1,000$ encodings of $X$ named $Z'$, $\mathbf{z} = \arg\min_{\mathbf{z}' \in Z'} \mathrm{dist}(h(\mathbf{z}'), X)$.

***Neighborhood Generators.*** For the counterfactual search and neighborhood generation, we experiment with four alternatives for the *neighgen* and *sample* functions of Algorithm 1 and Algorithm 2: *(i) Gaussian* (GA) that samples a normal distribution around $\mathbf{z}$ scaling each vector by a factor $\theta$ as detailed in Section 3.4; *(ii) Gaussian-matched* (GM) that is a Gaussian-based sampling [2] that uses distribution matching transport maps to minimize the problem of distribution mismatch. As in Gaussian, it samples the distribution around $\mathbf{z}$ scaling each vector by a factor $\theta$ as detailed in Section 3.4; *(iii) Uniform Sphere* (US) that samples a uniform sphere distribution around $\mathbf{z}$ with radius $\theta$; and *(iv) Matched Uniform* (MU) that combines a Gaussian-matched search to find the closest counterfactual, and Uniform Sphere sampling to generate the neighborhood. In the first three cases, the counterfactual search and the neighborhood generation use the same function, i.e., *sample* = *neighgen*, while the last approach uses two different ones, i.e., *sample* ≠ *neighgen*.

For the counterfactual search, we generate $n_s = 10{,}000$ instances at each iteration, starting with a threshold $\theta = 2$. Once the closest counterfactual is found, the neighborhood generation function *neighgen* is run with neighborhood size equal to $N = 500$ latent instances. If $\theta$ represents a radius, we perform the final sampling using as $\theta$ the distance between $\mathbf{z}$ and $\mathbf{z}_{\neq}$ while if $\theta$ represents a scaling factor, we take the last $\theta$ used in the counterfactual search. In the last generation step, we impose a balance between instance labels by oversampling the minority class. Moreover, we compare these sampling functions against the genetic approach adopted in [37]. For the genetic approach, we also generate $N = 500$ latent instances using the same parameters as in [37], namely 10 generations, *normalized Euclidean* distance as the genetic fitness function, probability of mutation equal to 0.5, probability of crossover equal to 0.7.

***Subsequence-based Surrogates.*** We implement the function $\varsigma^*$ of Algorithm 1 in two ways in order to test two different strategies to retrieve subsequences: SAX-based and shapelet-based. To extract SAX-based subsequences, we use the SAX-SEQL algorithm illustrated in [57], which extracts the $p$-most discriminative subsequences $S$ for $\hat{\mathbf{y}}$, and then converts time series in a binary-valued matrix $T$ using the symbolic subsequence transform detailed in Section 3.1. Furthermore, to extract *shapelets*-based subsequences, we adopt the LTS algorithm described in [30] that learns the $p$-most discriminative shapelets $S$ with respect to $\hat{\mathbf{y}}$ via gradient descent. The time series dataset is then converted to a simplified representation via the shapelet transform detailed in Section 3.1. In

Table 2. Fidelity Comparison (Higher is Better)

| | MODEL | GEN-SHP | GA-SHP | GA-SAX | GM-SHP | GM-SAX | MU-SHP | MU-SAX | US-SHP | US-SAX | GLO-SHP | GLO-SAX |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CBF | RES | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0.94 |
| | KNN | 1.00 | 1.00 | 1.00 | 0.97 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0.97 |
| COF | RES | 1.00 | 0.93 | 0.93 | 0.93 | 1.00 | 0.93 | 1.00 | 1.00 | 1.00 | 0.75 | 0.89 |
| | KNN | 1.00 | 0.93 | 1.00 | 0.93 | 1.00 | 0.96 | 1.00 | 1.00 | 1.00 | 0.89 | 0.89 |
| EC2 | RES | 0.98 | 0.94 | 1.00 | 0.96 | 0.99 | 0.98 | 1.00 | 0.95 | 0.99 | 0.69 | 0.78 |
| | KNN | 0.97 | 0.96 | 0.97 | 0.90 | 0.99 | 0.95 | 0.98 | 0.97 | 0.98 | 0.83 | 0.83 |
| GUN | RES | 0.98 | 0.82 | 0.98 | 0.78 | 0.98 | 0.84 | 0.98 | 0.80 | 0.98 | 0.93 | 0.90 |
| | KNN | 0.96 | 0.86 | 0.96 | 0.82 | 0.98 | 0.86 | 0.98 | 0.86 | 0.98 | 0.74 | 0.81 |
| $\overline{rk}$ | | 4.38 | 7.31 | 4.44 | 8.50 | **3.31** | 6.69 | **3.25** | 5.88 | **3.50** | 8.81 | 9.94 |

Top 3 performing models in bold.

order to have the same representation as in the SAX-subsequences alternative, the distances in $T$ are replaced with binary values using a threshold $\tau$ such that $\forall t_{i,j} \in T$ if the distance $t_{i,j} < \tau$, $t_{i,j}$ is replaced with 1 else with a 0. The distance threshold $\tau$ is chosen via grid search by testing the accuracy of the decision tree surrogate for every decile of the distribution of all the distances in $T$. To further simplify the classification task of the local surrogate, we binarize the label vector by considering only the predicted class of the instance to explain as 1 and all the others as 0. Therefore, $\forall \hat{y}_i \in \hat{\mathbf{y}}$, if $\hat{y}_i = b(X)$, $\hat{y}_i$ is replaced with a 1 else with a 0.

## 5.3 LASTS Framework Alternatives Analysis

In the following, we analyze the effect of various alternatives in terms of neighborhood generation and subsequence types on the explanation returned by LASTS.

We measure the performance in terms of fidelity, aiming to evaluate how good the explanation model is at mimicking the black-box decisions. The fidelity (*fid*) is simply defined as the accuracy between the prediction of the black-box and that of the explanation model [9]. In our setting, we compare the prediction of the black-box for the explanation dataset, $\hat{\mathbf{y}} = b(\mathcal{X}_{\text{exp}})$, and $\hat{\mathbf{y}}' = \{\hat{y}' | \forall X \in \mathcal{X}_{\text{exp}},\ \hat{y}' = dt(\varsigma^*(h(g(X))))\}$, where $dt$ is the local subsequence-based decision tree learned for each $X$ processed by LASTS. Formally, the fidelity is the percentage of times in which $\hat{\mathbf{y}} = \hat{\mathbf{y}}'$, i.e., $fid = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\hat{\mathbf{y}}=\hat{\mathbf{y}}'}$.

In [37], we showed that LASTS is more faithful than a version of LASTS using the same *random* neighborhood generation adopted by LIME [78]. In the following, we compare the version of LASTS proposed in [37] using the genetic-based neighborhood generation (named GEN-SHP) with the alternatives of LASTS proposed in this paper by combining the various sampling strategies for the neighborhood generation and types of subsequences. Besides, we show that extracting an explanation from the local neighborhood of a given instance is a winning strategy compared to an approach that builds a single global interpretable surrogate. Thus, we compare LASTS against shapelet/SAX-based global decision tree (*gdt*) classifiers, namely GLO-SHP and GLO-SAX, trained on $\mathcal{X}_{\text{bb}}$. In this case, the fidelity is calculated as the accuracy between $\hat{\mathbf{y}} = b(\mathcal{X}_{\text{exp}})$ and $\hat{\mathbf{y}}' = gdt(\mathcal{X}_{\text{exp}})$.

To help readability, we provide the average ranking ($\overline{rk}$) (lower is better) at the bottom of each table, which is the average of the ranks of each method for all datasets and black-boxes. Table 2 reports the values of the *fidelity*. We observe very high fidelity for all the approaches.

Table 3. Comparison of Precision, Coverage, and Silhouette for Alternative Neighborhood Generations and Types of Subsequences for the Explanation Returned by LASTS for RES (Higher is Better)

| | metric | GA-SHP | GA-SAX | GM-SHP | GM-SAX | MU-SHP | MU-SAX | US-SHP | US-SAX |
|---|---|---|---|---|---|---|---|---|---|
| | $pre_{r_=}$ | 0.998 | 1.000 | 0.999 | 1.000 | 0.999 | 1.000 | 0.999 | 1.000 |
| | $pre_{r_\neq}$ | 0.984 | 1.000 | 0.994 | 1.000 | 0.984 | 1.000 | 0.990 | 1.000 |
| CBF | $cov_{r_=}$ | 0.482 | 0.500 | 0.476 | 0.500 | 0.481 | 0.499 | 0.470 | 0.500 |
| | $cov_{r_\neq}$ | 0.392 | 0.375 | 0.340 | 0.375 | 0.423 | 0.472 | 0.379 | 0.445 |
| | $sil$ | 0.306 | 0.306 | 0.303 | 0.303 | 0.342 | 0.342 | 0.342 | 0.342 |
| | $pre_{r_=}$ | 0.870 | 0.998 | 0.866 | 0.998 | 0.884 | 0.998 | 0.912 | 0.996 |
| | $pre_{r_\neq}$ | 0.833 | 0.967 | 0.872 | 0.897 | 0.889 | 0.947 | 0.876 | 0.975 |
| COF | $cov_{r_=}$ | 0.476 | 0.441 | 0.498 | 0.458 | 0.458 | 0.482 | 0.460 | 0.481 |
| | $cov_{r_\neq}$ | 0.368 | 0.058 | 0.372 | 0.039 | 0.358 | 0.039 | 0.389 | 0.076 |
| | $sil$ | 0.206 | 0.206 | 0.208 | 0.208 | 0.228 | 0.228 | 0.229 | 0.229 |
| | $pre_{r_=}$ | 0.871 | 0.994 | 0.879 | 0.993 | 0.890 | 0.994 | 0.878 | 0.992 |
| | $pre_{r_\neq}$ | 0.829 | 0.920 | 0.838 | 0.882 | 0.828 | 0.888 | 0.842 | 0.885 |
| EC2 | $cov_{r_=}$ | 0.402 | 0.379 | 0.398 | 0.392 | 0.386 | 0.389 | 0.384 | 0.401 |
| | $cov_{r_\neq}$ | 0.306 | 0.022 | 0.287 | 0.036 | 0.326 | 0.022 | 0.305 | 0.025 |
| | $sil$ | 0.147 | 0.147 | 0.145 | 0.145 | 0.160 | 0.160 | 0.160 | 0.160 |
| | $pre_{r_=}$ | 0.863 | 0.999 | 0.860 | 0.998 | 0.884 | 0.999 | 0.888 | 0.999 |
| | $pre_{r_\neq}$ | 0.886 | 0.956 | 0.881 | 0.935 | 0.862 | 0.951 | 0.883 | 0.989 |
| GUN | $cov_{r_=}$ | 0.490 | 0.469 | 0.479 | 0.470 | 0.474 | 0.475 | 0.495 | 0.476 |
| | $cov_{r_\neq}$ | 0.434 | 0.083 | 0.472 | 0.106 | 0.467 | 0.112 | 0.443 | 0.096 |
| | $sil$ | 0.289 | 0.289 | 0.284 | 0.284 | 0.319 | 0.319 | 0.322 | 0.322 |
| | $pre_{r_=}$ | 7.500 | **2.000** | 7.000 | **2.875** | 5.750 | **2.000** | 5.750 | 3.125 |
| | $pre_{r_\neq}$ | 6.875 | **1.875** | 6.250 | 3.625 | 7.125 | **2.625** | 5.750 | **1.875** |
| $\overline{rk}$ | $cov_{r_=}$ | **3.000** | 6.500 | **3.500** | 4.875 | 6.125 | 4.000 | 5.250 | **2.750** |
| | $cov_{r_\neq}$ | **3.250** | 7.000 | 3.750 | 6.250 | **2.500** | 5.250 | **3.000** | 5.000 |
| | $sil$ | 6.000 | 6.000 | 7.000 | 7.000 | **3.000** | **3.000** | **2.000** | **2.000** |

Best 3 models for each metric in bold.

However, the null hypothesis that all methods are equivalent is rejected ($p\!-\!value < 0.01$) from the non-parametric Friedman over multiple datasets and black-boxes. We notice that the global approaches GLO-SHP and GLO-SAX always have slightly lower values than the local alternatives of LASTS. Among them, the variants employing SAX-based *dt* generally have higher fidelity than those using shapelet-based *dt*. Concerning the neighborhood generation options, results are very similar even though MU seem to return slightly better results. We better elaborate on this fact in the following paragraph.

The following test is to estimate the *precision* and *coverage* of the factual and counterfactual rules (indicated as $pre_{r_=}$, $pre_{r_\neq}$, $cov_{r_=}$, $cov_{r_\neq}$, respectively). The coverage is measured as the relative number of time series that respect the factual/counterfactual rule premises in the neighborhood $\hat{\mathcal{X}}$, while the precision is measured as the relative number of covered time series for which the prediction outcome is correct. In addition, we measure also the *cohesion* and *separation* of the neighborhoods $\hat{\mathcal{X}}$ through the *silhouette* (*sil*) coefficient [85] with respect to the two clusters identified as $\hat{\mathcal{X}}_=$ and $\hat{\mathcal{X}}_\neq$ [32]. Table 3 illustrates the average precision, coverage, and silhouette scores

Table 4. Mean Running Time in Seconds (Lower is Better)

| | | CBF | COF | EC2 | GUN |
|---|---|---|---|---|---|
| *neighgen* | CFS | 4.83 ± 0.18 | 8.18 ± 0.44 | 6.81 ± 0.54 | 7.47 ± 0.63 |
| | GENETIC | 55.57 ± 4.55 | 80.67 ± 2.11 | 72.25 ± 5.14 | 80.81 ± 18.00 |
| $\varsigma^*$ | SHP | 60.02 ± 5.43 | 46.19 ± 5.24 | 47.33 ± 8.53 | 42.25 ± 7.85 |
| | SAX | 8.91 ± 0.47 | 43.75 ± 9.33 | 13.04 ± 2.61 | 12.63 ± 2.14 |
| *all* | GEN-SHP | 115.59 ± 9.98 | 126.86 ± 7.35 | 119.58 ± 13.67 | 123.06 ± 9.83 |
| | MU-SHP tot | 64.85 ± 5.61 | 54.38 ± 5.69 | 54.14 ± 9.07 | 49.72 ± 8.48 |
| | MU-SAX tot | **13.74 ± 0.65** | **51.93 ± 9.77** | **19.85 ± 3.15** | **20.10 ± 2.77** |

Best results in bold.

when applying LASTS for explaining the RES classifier. For every metric, the higher the value, the better the score. Again, we notice that SAX-based methods seem to perform better than shapelet-based ones. More in detail, MU-SAX has always the highest $pre_{r_=}$, while GA-SAX is among the bests for $pre_{r_\neq}$. Factual coverage scores are, in general, quite close, while shapelet-based models usually have better counterfactual coverage. Finally, for the silhouette, the approaches using the uniform sphere *neighgen* function (MU and US) perform better by quite a margin. In summary, we can state that LASTS explanations obtained using SAX and uniform sphere or matched uniform are the best ones. However, in the following, we only consider LASTS using SAX-based subsequences and the MU neighborhood generation, unless otherwise specified, because the GM sampling does not guarantee a perfectly centered neighborhood generation around the closest counterexemplar. This problem can be detected by computing the **Local Outlier Factor (LOF)** [10] for **z**, as implemented in `sklearn`. In particular, a LOF score of −1 indicates an outlier, while a score of 1 indicates an inlier. In our tests $LOF_{\text{MU}} = 1$ for every dataset and for both black-boxes, while $LOF_{\text{GM}} \in [-0.55, 0.64]$, indicating a higher degree of outlierness. Moreover, given that the radius of the uniform sampling is, by construction, as small as possible, the problem of distribution mismatch in a Gaussian space is minimized in most cases.

The time required by LASTS for the explanation is mainly affected by the CFS algorithm, and the subsequence transform ($\varsigma^*$). The CFS algorithm's most expensive operations are the black-box ($b$) and decoder ($h$) functions. They dominate the time complexity of all the *sample* functions described in Section 4.3, which is only $O(q)$, where $q$ is the dimensionality of the latent vector $\mathbf{z}$ [88]. Following Algorithm 2, $b$ and $h$ must be repeated until no counterexemplar in the latent space is found. Therefore, the worst-case scenario can appear when the latent encoding of $\mathbf{z}$ is extremely close to the decision boundary, and we have to reduce the $\theta$ parameter a great number of times. This is extremely unlikely, and we found that in most of our tests, starting with $\theta = 2$, only a few iterations are needed to exit the loop, with the worst case not exceeding 20 iterations. In Table 4, we report the average *runtime* and standard deviation for the two critical phases and for the complete explanation process (*all*). We notice that on every dataset, the CFS strategy is one order of magnitude faster than the genetic one[4]. The subsequence extraction runtime is comparable only for the Coffee dataset but, on average, is at least four times faster when using SAX instead of shapelets. In conclusion, we notice that the complete explanation process is much faster in the novel version of LASTS when using SAX with respect to the one proposed in [37] and the current one when using shapelets.

[4]We adopted Gaussian-matched sampling for this experiment for the CFS strategy. However, different sampling methods do not significantly affect performance.

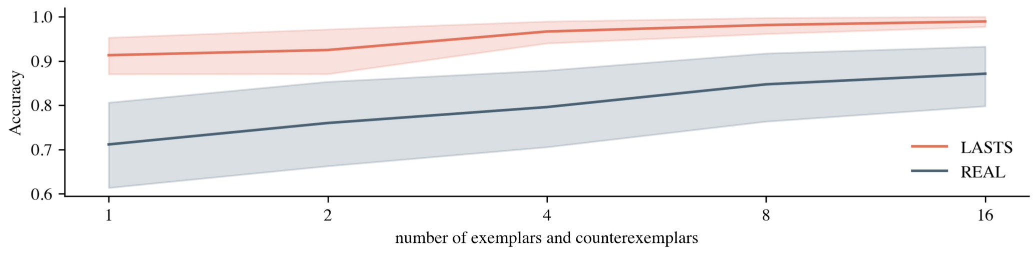

Fig. 7. Usefulness benchmark. Mean and 95% confidence intervals of the 1-NN accuracy for all datasets, varying the number of exemplars and counterexemplars used for training.

In summary, MU-SAX is the most well-rounded approach: *(i)* it generates a neighborhood of well-separated exemplar and counterexemplar instances around the decision boundary, which contains **z** as an inlier, *(ii)* it has excellent performance as a local surrogate, with high fidelity scores, factual and counterfactual rules precision, and *(iii)* it ensures that the closest counterexemplar is sampled from a "probable" portion of latent space, minimizing the distribution mismatch.

## 5.4 Instance-based Explanation Experiments

Since it is hard to validate the *usefulness* of the generated factual and counterfactual instances with an experiment involving humans, inspired by [50], we tested their effectiveness with a memory-based machine learning technique. This experiment gives an objective and indirect estimation of the usefulness of exemplars and counterexemplars by checking how the performance of a simple classifier changes by training it on an increasing number of instances. For each instance $X \in \mathcal{X}_{\text{exp}}$ we apply LASTS, i.e., we encode $X$, find its closest counterexemplar, generate its latent neighborhood, and decode it into $\hat{\mathcal{X}}$. From this synthetic dataset $\hat{\mathcal{X}}$, using the black-box to retrieve the predicted labels, we extract $n$ exemplars, i.e., $n$ instances having the same class as $X$, and $n$ counterexemplars from each other class, i.e., $n(c-1)$ instances having a different class w.r.t. $X$. This extraction is random and without replacement. Then, we use the selected exemplars and counterexemplars to train a **1-Nearest Neighbor (1-NN)**, and we classify $X$. For each dataset, we compare the classification performances on $X$ with a 1-NN trained on *real* time series, $n$ per class, randomly selected from $\mathcal{X}_{\text{exp}} \setminus \{X\}$. Specifically, $n \in \{1, 2, 4, 8, 16\}$ and the Euclidean distance is used as a distance function. Figure 7 shows the accuracy of a 1-NN using an increasing number of exemplars and counterexemplars as training. The plot aggregates the accuracy for each dataset at each $n$, showing the estimate of the central tendency and the respective confidence interval. On average, the performance of LASTS is higher and more constant, revealing that few exemplars and counterexemplars are a good proxy for recognizing the classification outcome. On the other hand, the accuracy of REAL increases more steeply on the various datasets with increasing $n$, meaning that more exemplars and counterexemplars are necessary to distinguish the reasons for the classification. Hence, we can state that exemplars and counterexemplars help in discovering the decision boundary and in highlighting similarities and differences. This experiment shows that time series must be carefully chosen, as LASTS does, in order to be suitable for class recognition.

## 5.5 Saliency-based Explanation Experiments

In these experiments, we compare LASTS with SHAP [64] employed to solve the time series black-box outcome explanation problem as in [4]. Similar to the proposal in [71], we adapt SHAP to time series classifiers by performing an *adaptive* segmentation [49] of the time series in order to divide it into meaningful intervals that are used as features by SHAP. As is commonly done with images,

Table 5. Saliency Map Deletion (Lower is Better), Insertion (Higher is Better), Stability (Lower is Better)

| | *deletion* ↓ | | *insertion* ↑ | | *stability* ↓ | |
|---|---|---|---|---|---|---|
| | LASTS | SHAP | LASTS | SHAP | LASTS | SHAP |
| ART | 0.30 ± 0.20 | 0.30 ± 0.26 | 0.35 ± 0.23 | 0.30 ± 0.22 | 0.16 ± 0.04 | 0.32 ± 0.04 |
| CBF | 0.78 ± 0.17 | 0.75 ± 0.23 | 0.72 ± 0.25 | 0.78 ± 0.28 | 0.04 ± 0.08 | 0.20 ± 0.17 |
| CBM | 0.82 ± 0.08 | 0.79 ± 0.18 | 0.90 ± 0.11 | 0.73 ± 0.20 | 0.04 ± 0.04 | 0.31 ± 0.09 |
| COF | 0.47 ± 0.50 | 0.51 ± 0.46 | 0.47 ± 0.50 | 0.48 ± 0.49 | 0.09 ± 0.03 | 0.16 ± 0.09 |
| EC2 | 0.74 ± 0.37 | 0.50 ± 0.26 | 0.79 ± 0.28 | 0.92 ± 0.17 | 0.11 ± 0.06 | 0.27 ± 0.13 |
| EC5 | 0.64 ± 0.37 | 0.71 ± 0.22 | 0.93 ± 0.10 | 0.93 ± 0.11 | 0.03 ± 0.03 | 0.27 ± 0.13 |
| ERI | 0.43 ± 0.30 | 0.50 ± 0.30 | 0.41 ± 0.29 | 0.39 ± 0.31 | 0.19 ± 0.08 | 0.33 ± 0.05 |
| GUN | 0.64 ± 0.44 | 0.62 ± 0.46 | 0.61 ± 0.48 | 0.60 ± 0.47 | 0.16 ± 0.06 | 0.30 ± 0.20 |
| ITA | 0.50 ± 0.38 | 0.65 ± 0.30 | 0.83 ± 0.19 | 0.73 ± 0.26 | 0.07 ± 0.08 | 0.27 ± 0.25 |
| LIB | 0.33 ± 0.23 | 0.26 ± 0.22 | 0.36 ± 0.23 | 0.22 ± 0.18 | 0.23 ± 0.06 | 0.35 ± 0.07 |
| PEN | 0.51 ± 0.24 | 0.46 ± 0.18 | 0.41 ± 0.23 | 0.46 ± 0.16 | 0.21 ± 0.10 | 0.38 ± 0.15 |
| PHA | 0.32 ± 0.41 | 0.48 ± 0.21 | 0.30 ± 0.42 | 0.40 ± 0.24 | 0.21 ± 0.05 | 0.32 ± 0.22 |
| PLA | 0.40 ± 0.28 | 0.40 ± 0.33 | 0.36 ± 0.36 | 0.45 ± 0.27 | 0.06 ± 0.06 | 0.17 ± 0.05 |
| STR | 0.55 ± 0.48 | 0.54 ± 0.46 | 0.55 ± 0.48 | 0.58 ± 0.44 | 0.14 ± 0.07 | 0.11 ± 0.11 |
| TWO | 0.49 ± 0.47 | 0.59 ± 0.39 | 0.59 ± 0.38 | 0.57 ± 0.42 | 0.10 ± 0.09 | 0.27 ± 0.13 |
| $\overline{rk}$ | 1.53 | **1.47** | 1.53 | **1.47** | **1.07** | 1.93 |

the absence of a feature is simulated via linear interpolation by connecting the observations before and after the ablated segment.

A common strategy to validate a saliency-based explanation is to observe how the performance of the black-box changes by adding/removing features in order of importance [76]. Regarding *deletion*, the intuition is that removing the most important time-steps in a time series will force the black-box to change its decision. On the other hand, the *insertion* evaluation adopts a complementary approach by starting from an “empty” time series and adding the most important time-steps. From a practical standpoint, the removal of time-steps is approximated by taking the average of the time series values. Time-steps are added/deleted in order of importance one by one, and the black-box prediction is checked at each step. Insertion/deletion metrics are computed as **AUC**/$md$, i.e., the **Area Under the Curve** of the line obtained by computing the accuracy of the black-box after each insertion/deletion, divided by the total number of time-steps $m$ of every dimension $d$ of the time series. For the insertion benchmark, we want the black-box performance to improve as fast as possible; therefore, a high score is desirable. On the contrary, we expect a sudden drop in performance for deletion benchmarks, resulting in a lower score. Results can be seen in Table 5. For this benchmark, LASTS scores very similarly to SHAP, indicating that both methods are able to indicate important observations of the time series.

Furthermore, we measure the *stability* of an explainer as its ability to produce a similar explanation for close instances, i.e., given similar instances, their saliency map should also be similar. In practice, given an instance $X$, we find its closest instance $X'$ in the latent space using the Euclidean distance. Then, we compare their saliency maps using the **Mean Absolute Error (MAE)**. Intuitively, stable explanations should have lower MAE, while unstable explanations should have a higher error. As can be seen in Table 5, LASTS outperforms SHAP in stability for all but one dataset, indicating its ability to give similar explanations for similar instances.

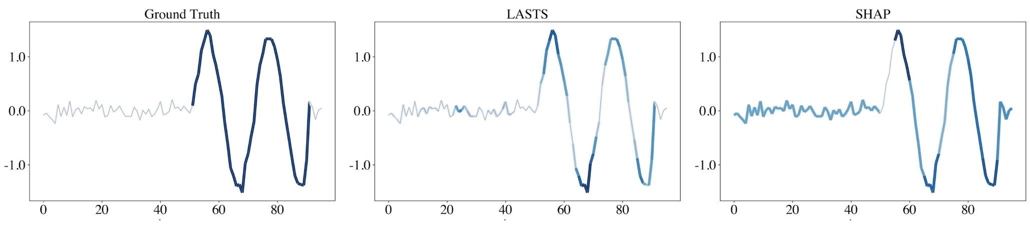

Fig. 8. (*left*): ground truth of a synthetic model that classifies a time series using the highlighted blue sinusoidal pattern. (*center*): saliency map returned by LASTS. (*right*): saliency map returned by SHAP.

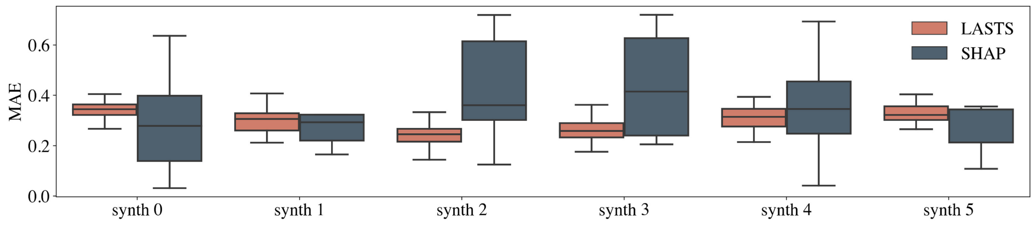

Fig. 9. Box-plots of correctness measured as MAE (lower is better).

Since it is impossible to know exactly if explanations are correct with real datasets and standard black-box models, we use a synthetic experiment to check if the saliency maps obtained with agnostic explainers match with custom-defined ground truths, i.e., to check if they are correct. To perform this study, inspired by [31], we generate a univariate synthetic dataset having two classes. Time series belonging to each class have a distinct and easily identifiable pattern that unequivocally defines their class, plus some random noise. Then, we build a synthetic classifier that performs the classification by only looking at the presence of these predefined patterns. The time-steps of the time series that are checked by the synthetic model during classification are thus known, and the ground truth is defined as a vector of length $m$ with the value 1 if the pattern is present at a given time-step and 0 otherwise. In other words, the only important points for the time series classification have a value of 1, while the noise has a 0 value. Given a saliency-based explanation returned as a vector of length $m$, we compare it with the ground truth by first normalizing it in the range $[0, 1]$ and then computing MAE between the ground truth and the normalized saliency vector. Intuitively, if the saliency vector correctly identifies important and irrelevant points, the MAE will tend to 0. To improve the test's significance, we perform it on six synthetic classifiers that use different random and continuous subsets of the original patterns. A comparison of the saliency maps returned by LASTS and SHAP w.r.t. the ground truth is depicted in Figure 8, while box-plots of the MAE for each dataset can be viewed in Figure 9. In general, LASTS performs better, i.e., has a lower median MAE in four of the six tests. Moreover, LASTS has a lower standard deviation, indicating that even its worst saliency maps are not that far from the ground truth. In general, we observe that LASTS tends to give a more targeted and precise explanation, while SHAP tends to give importance to more points in the time series, resulting in a very wide interquartile range for the MAE.

## 5.6 Rule-based Explanation Benchmarks

To the best of our knowledge, LASTS is the only local agnostic time series explainer that outputs rules as an explanation. Thus, we decided to compare the factual and counterfactual rules returned by LASTS with those of a global decision tree, as in Section 5.3, and also with ANCHOR [79]. In order

Table 6. Precision (Higher is Better), Coverage (Higher is Better), and Length (Lower is Better) of Factual and Counterfactual Rules

| | $pre_{r_=}$ ↑ | | | $cov_{r_=}$ ↑ | | | $len_{r_=}$ ↓ | | | $pre_{r_\neq}$ ↑ | | $cov_{r_\neq}$ ↑ | | $len_{r_\neq}$ ↓ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | LASTS | GLO-SAX | ANCHOR | LASTS | GLO-SAX | ANCHOR | LASTS | GLO-SAX | ANCHOR | LASTS | GLO-SAX | LASTS | GLO-SAX | LASTS | GLO-SAX |
| ART | 0.92 | 0.66 | - | 0.30 | 0.04 | - | 4.70 | 10.36 | - | 0.78 | 0.54 | 0.09 | 0.04 | 4.92 | 11.04 |
| CBF | 1.00 | 0.89 | 1.00 | 0.50 | 0.33 | 0.20 | 1.14 | 1.69 | 2.00 | 0.99 | 0.86 | 0.43 | 0.33 | 1.22 | 2.00 |
| CBM | 0.99 | 0.97 | 1.00 | 0.46 | 0.33 | 0.18 | 1.33 | 1.67 | 2.00 | 0.96 | 0.95 | 0.38 | 0.33 | 1.78 | 2.00 |
| COF | 1.00 | 0.93 | 1.00 | 0.45 | 0.50 | 0.21 | 3.64 | 1.00 | 2.00 | 0.93 | 0.95 | 0.01 | 0.50 | 3.96 | 1.00 |
| EC2 | 0.99 | 0.84 | 1.00 | 0.43 | 0.36 | 0.13 | 3.84 | 1.96 | 2.36 | 0.94 | 0.65 | 0.02 | 0.15 | 4.24 | 3.16 |
| EC5 | 1.00 | 0.98 | 1.00 | 0.48 | 0.47 | 0.22 | 2.34 | 5.10 | 2.00 | 0.98 | 1.00 | 0.11 | 0.01 | 2.80 | 5.14 |
| ERI | 0.98 | 0.58 | 1.00 | 0.43 | 0.17 | 0.06 | 3.00 | 3.72 | 3.04 | 0.89 | 0.60 | 0.10 | 0.17 | 3.50 | 4.20 |
| GUN | 0.99 | 0.94 | 1.00 | 0.44 | 0.50 | 0.13 | 2.90 | 1.00 | 2.44 | 0.94 | 0.93 | 0.06 | 0.50 | 3.56 | 1.00 |
| ITA | 0.99 | 0.80 | 1.00 | 0.46 | 0.37 | 0.19 | 2.38 | 2.48 | 2.14 | 0.94 | 0.54 | 0.12 | 0.11 | 3.26 | 2.58 |
| LIB | 0.94 | 0.72 | 0.97 | 0.26 | 0.05 | 0.02 | 4.64 | 5.78 | 11.70 | 0.79 | 0.88 | 0.05 | 0.02 | 5.02 | 6.12 |
| PEN | 0.94 | 0.88 | 0.99 | 0.33 | 0.02 | 0.04 | 2.46 | 10.26 | 4.12 | 0.85 | 0.25 | 0.17 | <0.01 | 3.44 | 10.76 |
| PHA | 0.91 | 0.76 | 1.00 | 0.30 | 0.23 | 0.08 | 3.58 | 9.50 | 2.56 | 0.81 | 0.77 | 0.09 | 0.01 | 4.26 | 10.02 |
| PLA | 1.00 | 0.92 | 1.00 | 0.49 | 0.13 | 0.08 | 1.98 | 4.32 | 3.54 | 0.99 | 0.91 | 0.12 | 0.12 | 2.28 | 4.92 |
| STR | 0.99 | 0.90 | 1.00 | 0.38 | 0.43 | 0.12 | 3.76 | 2.02 | 2.48 | 0.91 | 0.84 | 0.03 | 0.05 | 4.50 | 2.50 |
| TWO | 1.00 | 0.96 | 1.00 | 0.43 | 0.50 | 0.10 | 2.06 | 1.00 | 2.92 | 0.93 | 0.96 | 0.15 | 0.50 | 2.96 | 1.00 |
| $\overline{rk}$ | 1.90 | 2.93 | **1.04** | **1.27** | 1.80 | 2.93 | **1.80** | 2.07 | 2.07 | **1.27** | 1.73 | **1.47** | 1.53 | **1.40** | 1.60 |

to adapt ANCHOR to time series, we consider each observation as a separate feature. As a note, ANCHOR can return only factual rules whose conditions depend on single time series observations. We use precision and coverage metrics to evaluate the goodness of a rule. Moreover, given that simpler explanations are to be preferred, we also measure the length of the returned rules.

The results of *precision*, *coverage* and *length* for the factual rules returned by LASTS, GLO-SAX and ANCHOR are presented in Table 6. Regarding precision, ANCHOR is the clear winner in all but one dataset. This result is not surprising given that ANCHOR, by definition, constructs rules that are guaranteed to have a precision above 0.95. LASTS scores slightly lower, with a precision that does not drop under 0.9, while GLO-SAX is the clear loser in this benchmark. The coverage metric helps to show the whole picture of these benchmarks. LASTS performs best, followed by GLO-SAX and ANCHOR in last place. This indicated that while rules returned by ANCHOR are indeed slightly more precise, they are also less generalizable, i.e., they cover a much lower number of instances. Furthermore, as shown by the average lengths, factual rules returned by ANCHOR and GLO-SAX are also considerably longer, i.e., more difficult to understand from a human standpoint. As a note, ANCHOR is also extremely inefficient for longer time series, requiring hours of runtime to explain a single instance. For this reason, completing the benchmarks for the ART dataset was impossible. Regarding counterfactual rules, results are presented in Table 6. LASTS performs better than a global surrogate in precision and length, tying in coverage. This experiment demonstrates that, while the number of instances covered by the rules is comparable for the two methods, counterfactual rules returned by LASTS are more precise, shorter, and thus easier to understand.

## 5.7 Qualitative Examples

This section shows qualitative examples from two real-world datasets, ECG5000 and Libras. Figure 10 presents an explanation of an instance from the ECG5000 dataset. ECG5000 contains 5,000 heartbeats belonging to five different classes, one corresponding *Normal* instances, and four corresponding to different kinds of *Abnormal* heartbeats. The instance we are trying to explain is

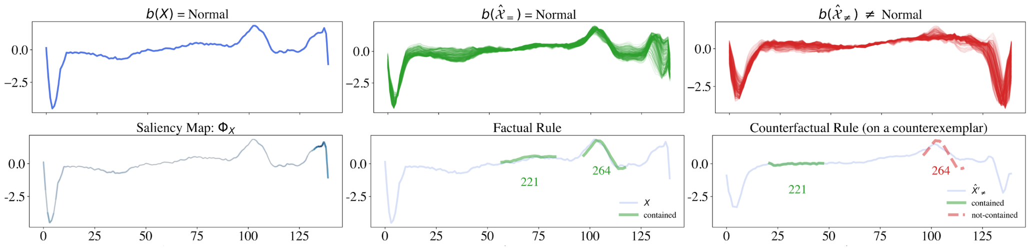

Fig. 10. Explanation of the prediction of ROCKET for an instance of the ECG5000 dataset. From left to right: (*top*) instance to explain, exemplars, counterexemplars, (*bottom*) saliency map, factual rule and counterfactual rule (shown over a counterexemplar).

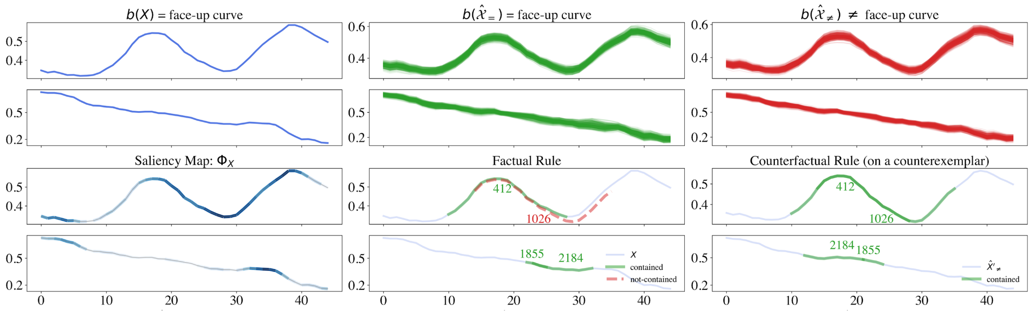

Fig. 11. Explanation of the prediction of ROCKET for an instance of the Libras dataset. From left to right: (*top*) instance to explain, exemplars, counterexemplars, (*bottom*) saliency map, factual rule and counterfactual rule (shown over a counterexemplar).

correctly classified by ROCKET as *Normal* (Figure 10 top-left). By looking at the difference between exemplars and counterexemplars, we can clearly see that the main difference between normal and abnormal instances is in the rightmost part of the time series, which is much lower for abnormal time series, and presents an evident V-shape. More specifically, these abnormal series belong all to the *Premature Ventricular Contraction* class. The saliency map confirms the assessment deduced by the example-based explanation, highlighting only the last observations of the time series. The rules show the other main difference between classes. The factual rule, $r_= = \{s_{221} \in X \wedge s_{264} \in X\} \rightarrow$ *Normal*, shows that normal instances contain subsequence $s_{264}$, while the counterfactual rule, $r_\neq = \{s_{221} \in X \wedge s_{264} \notin X\} \rightarrow$ *Premature Ventricular Contraction*, shows that abnormal time series have a flatter shape, not containing $s_{264}$. In general, we do not expect the saliency map and the rules to cover the same exact areas because the saliency map only highlights the parts of the time series that change the most between classes, whereas the rules can be based on subsequences emphasizing even a small shape change from any part of the time series.

In Figure 11, we present an explanation of a multivariate time series from the Libras dataset. This dataset contains instances having two signals each, belonging to 15 classes that correspond to different hand movements. The instance we are trying to explain is labeled as *face-up curve*, and it is correctly classified by ROCKET. The example-based part of the explanation shows exemplars and counterexemplars that are extremely similar to the naked eye. This means that even very small changes in the time series shape can result in a change of prediction from the black-box. In this sense, the closest instances belong all to class *horizontal wavy*. The most salient observations are highlighted in the saliency map and show that the top signal, and in particular the central sigmoidal part, is the most relevant for the change in classification. This is also confirmed by the factual and counterfactual rules, $r_{=} = \{s_{412} \in \mathbf{x}_0 \wedge s_{1026} \notin \mathbf{x}_0 \wedge s_{1855} \in \mathbf{x}_1 \wedge s_{2184} \in \mathbf{x}_1\} \rightarrow$ *face-up curve*, $r_{\neq} = \{s_{412} \in \mathbf{x}_0 \wedge s_{1026} \in \mathbf{x}_0 \wedge s_{1855} \in \mathbf{x}_1 \wedge s_{2184} \in \mathbf{x}_1\} \rightarrow$ *horizontal wavy*. The rules show that the most significant subsequence is $s_{1026}$, given that its presence/absence results in a change in class. Figure 11 (bottom-right) shows a counterexemplar which contains $s_{1026}$, i.e., has a lower dip in the central sinusoidal pattern w.r.t. $X$.

## 6 CONCLUSIONS

We have presented LASTS, a local model-agnostic subsequence-based explainer that returns an easy-to-understand explanation for univariate and multivariate time series classifiers. LASTS succeeds in addressing the time series black-box outcome explanation problem, returning three different kinds of explanations: a saliency map, examples, and decision rules. The saliency map highlights the most important observations of the time series for the classification. Exemplar and counterexemplar instances can be compared to the time series to explain the black-box behavior. Finally, subsequence-based decision rules allow an understanding of the logic of the classification, showing the reasons for the outcome in terms of patterns that must and must not be contained. Extensive experimentation shows that LASTS outperforms existing explainers in returning meaningful, useful, faithful, and coherent explanations.

The proposed method has some limitations. Indeed, the subsequences-based rules do not consider multiple alignments of the same shapelet at different time series points. On the other hand, multiple occurrences could help better explain a predictive phenomenon. Also, technical and conceptual extensions are possible. First, we would like to test LASTS on longer and more complex, real-world time series datasets while also extending it to different types of sequential data like trajectories, text, and shopping transactions. Second, we would like to deepen the study of the relationship between the latent and subsequence spaces. Third, we aim to empower the explanations' expressiveness and to enable higher levels of abstraction with grammar-based decision trees [58]. Fourth, we also aim to explain the overall logic of a time series classifier by aggregating the local subsequence-based rules into a global explanation model [84]. Finally, a human decision-making task driven by LASTS explanations could objectively evaluate the real effectiveness of the explanations.

## APPENDIX

## A NOTATION

Table 7. Summary of Notation

| **Data** | |
|---|---|
| $\mathcal{X}, X, \mathbf{x}, x$ | time series dataset, instance, signal, observation |
| $\mathbf{y}, y, c$ | labels vector, value, number of unique labels |
| $n, i$ | number of instances in a dataset, instance index |
| $m, j$ | number of observations in a time series, feature index |
| $d, k$ | number of signals in a time series, signal index |
| $S, \mathbf{s}$ | collection of subsequences, subsequence |
| $p, l$ | number, length of extracted subsequences |
| **Models** | |
| $f(\cdot)$ | generic function |
| $b(\cdot)$ | black-box classifier |
| $rm(\cdot)$ | rule-based classifier |
| $dt(\cdot)$ | decision tree classifier |
| $\hat{\mathbf{y}}, \hat{y}$ | classifier prediction for a time series dataset, instance |
| **Transform** | |
| $\varsigma(\cdot), T$ | shapelet or subsequence transform, transformed dataset |
| $\tilde{\mathbf{x}}$ | SAX-transformed time series signal |
| $w$ | number of intervals of PAA |
| $\mathbb{A}$ | SAX alphabet |
| **Autoencoder** | |
| $g(\cdot), h(\cdot)$ | encoder, decoder |
| $q$ | number of latent dimensions |
| $\theta$ | latent vector scaling factor |
| $\mathbf{u}$ | randomly sampled normal Gaussian vector |
| $Z, \mathbf{z}$ | latent encoding of a time series dataset, instance |
| $\hat{\mathcal{X}}, \hat{X}$ | autoencoding of a time series dataset, instance |
| **Explanation** | |
| $E, e$ | human-interpretable domain, explanation |
| $\Phi, \phi$ | saliency map, saliency value |
| $Z_=, Z_{\neq}$ | exemplar, counterexemplar time series encodings |
| $\hat{\mathcal{X}}_=, \hat{\mathcal{X}}_{\neq}$ | exemplar, counterexemplar time series dataset, |
| $\mathbf{z}_{\neq}, \hat{X}_{\neq}$ | best counterfactual latent vector, time series |
| $r_=, r_{\neq}$ | factual, counterfactual rule |

## REFERENCES

[1] Amina Adadi and Mohammed Berrada. 2018. Peeking inside the black-box: A survey on Explainable Artificial Intelligence (XAI). *IEEE Access* 6 (2018), 52138–52160.

[2] Eirikur Agustsson, Sage Alexander, Radu Timofte, and Luc Van Gool. 2017. Optimal transport maps for distribution preserving operations on latent spaces of Generative Models. (11 2017).

[3] Andrea Apicella, Francesco Isgrò, Roberto Prevete, and Guglielmo Tamburrini. 2019. Contrastive explanations to classification systems using sparse dictionaries. In *Image Analysis and Processing - ICIAP 2019 - 20th International Conference, Trento, Italy, September 9–13, 2019, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 11751)*, Elisa Ricci, Samuel Rota Bulò, Cees Snoek, Oswald Lanz, Stefano Messelodi, and Nicu Sebe (Eds.). Springer, 207–218. https://doi.org/10.1007/978-3-030-30642-7_19

[4] Hiba Arnout, Mennatallah El-Assady, Daniela Oelke, and Daniel A. Keim. 2019. Towards a rigorous evaluation of XAI methods on time series. In *2019 IEEE/CVF International Conference on Computer Vision Workshops, ICCV Workshops 2019, Seoul, Korea (South), October 27–28, 2019*. IEEE, 4197–4201. https://doi.org/10.1109/ICCVW.2019.00516

[5] Emre Ates, Burak Aksar, Vitus J. Leung, and Ayse K. Coskun. 2021. Counterfactual explanations for multivariate time series. In *2021 International Conference on Applied Artificial Intelligence (ICAPAI)*. IEEE, 1–8.

[6] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. 2017. The great time series classification bake off: A review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery* 31, 3 (2017), 606–660.

[7] Rachana Balasubramanian, Samuel Sharpe, Brian Barr, Jason D. Wittenbach, and C. Bayan Bruss. 2020. Latent-CF: A simple baseline for reverse counterfactual explanations. *CoRR* abs/2012.09301 (2020). arXiv:2012.09301 https://arxiv.org/abs/2012.09301

[8] Gustavo E. A. P. A. Batista, Xiaoyue Wang, and Eamonn J. Keogh. 2011. A complexity-invariant distance measure for time series. In *Proceedings of the Eleventh SIAM International Conference on Data Mining, SDM 2011, April 28–30, 2011, Mesa, Arizona, USA*. SIAM / Omnipress, 699–710. https://doi.org/10.1137/1.9781611972818.60

[9] Francesco Bodria, Fosca Giannotti, Riccardo Guidotti, Francesca Naretto, Dino Pedreschi, and Salvatore Rinzivillo. 2023. Benchmarking and survey of explanation methods for black box models. *Data Min. Knowl. Discov.* 37, 5 (2023), 1719–1778. https://doi.org/10.1007/s10618-023-00933-9

[10] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. 2000. LOF: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16–18, 2000, Dallas, Texas, USA*, Weidong Chen, Jeffrey F. Naughton, and Philip A. Bernstein (Eds.). ACM, 93–104. https://doi.org/10.1145/342009.335388

[11] Romain Briandet, E. Katherine Kemsley, and Reginald H. Wilson. 1996. Discrimination of Arabica and Robusta in instant coffee by Fourier transform infrared spectroscopy and chemometrics. *Journal of Agricultural and Food Chemistry* 44, 1 (1996), 170–174. https://doi.org/10.1021/jf950305a

[12] Ruth M. J. Byrne. 2019. Counterfactuals in explainable artificial intelligence (XAI): Evidence from human reasoning. In *Proceedings of the Twenty-eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10–16, 2019*, Sarit Kraus (Ed.). ijcai.org, 6276–6282. https://doi.org/10.24963/ijcai.2019/876

[13] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. 2019. The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica* 6, 6 (2019), 1293–1305.

[14] Hoang Anh Dau and Eamonn J. Keogh. 2017. Matrix profile V: A generic technique to incorporate domain knowledge into motif discovery. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 – 17, 2017*. ACM, 125–134. https://doi.org/10.1145/3097983.3097993

[15] Luke M. Davis. 2013. *Predictive Modelling of Bone Ageing*. Ph. D. Dissertation. University of East Anglia, Norwich, UK. https://ueaeprints.uea.ac.uk/45085/

[16] Eoin Delaney, Derek Greene, and Mark T. Keane. 2021. Instance-based counterfactual explanations for time series classification. In *Case-Based Reasoning Research and Development*, Antonio A. Sánchez-Ruiz and Michael W. Floyd (Eds.). Springer International Publishing, Cham, 32–47.

[17] Angus Dempster, François Petitjean, and Geoffrey I. Webb. 2020. ROCKET: Exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery* 34, 5 (2020), 1454–1495.

[18] Angus Dempster, Daniel F. Schmidt, and Geoffrey I. Webb. 2021. MiniRocket: A very fast (almost) deterministic transform for time series classification. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14–18, 2021*, Feida Zhu, Beng Chin Ooi, and Chunyan Miao (Eds.). ACM, 248–257. https://doi.org/10.1145/3447548.3467231

[19] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Pai-Shun Ting, Karthikeyan Shanmugam, and Payel Das. 2018. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018,*

*NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). 590–601. https://proceedings.neurips.cc/paper/2018/hash/c5ff2543b53f4cc0ad3819a36752467b-Abstract.html

[20] Daniel B. Dias, Renata C. B. Madeo, Thiago Rocha, Helton Hideraldo Bíscaro, and Sarajane Marques Peres. 2009. Hand movement recognition for Brazilian Sign Language: A study using distance-based neural networks. In *International Joint Conference on Neural Networks, IJCNN 2009, Atlanta, Georgia, USA, 14–19 June 2009*. IEEE Computer Society, 697–704. https://doi.org/10.1109/IJCNN.2009.5178917

[21] JGA Dolfing, EHL Aarts, and JJGM van Oosterhout. 1998. Combining multiple classifiers for pen-based handwritten digit recognition. In *Proceedings of the Fourteenth International Conference on Pattern Recognition*, Vol. 2. 1309–1312.

[22] Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608* (2017).

[23] Johann Faouzi and Hicham Janati. 2020. pyts: A python package for time series classification. *Journal of Machine Learning Research* 21, 46 (2020), 1–6. http://jmlr.org/papers/v21/19-763.html

[24] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. 2018. Data augmentation using synthetic data for time series classification with deep residual networks. *CoRR* abs/1808.02455 (2018). arXiv:1808.02455 http://arxiv.org/abs/1808.02455

[25] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. 2019. Deep learning for time series classification: A review. *Data Mining and Knowledge Discovery* 33, 4 (2019), 917–963.

[26] Michael Flynn, James Large, and Tony Bagnall. 2019. The contract random interval spectral ensemble (c-RISE): The effect of contracting a classifier on accuracy. In *Hybrid Artificial Intelligent Systems - 14th International Conference, HAIS 2019, León, Spain, September 4–6, 2019, Proceedings (Lecture Notes in Computer Science, Vol. 11734)*, Hilde Pérez García, Lidia Sánchez-González, Manuel Castejón Limas, Héctor Quintián-Pardo, and Emilio S. Corchado Rodríguez (Eds.). Springer, 381–392. https://doi.org/10.1007/978-3-030-29859-3_33

[27] Ary L. Goldberger, Luis A. N. Amaral, Leon Glass, Jeffrey M. Hausdorff, Plamen Ch Ivanov, Roger G. Mark, Joseph E. Mietus, George B. Moody, Chung-Kang Peng, and H. Eugene Stanley. 2000. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation* 101, 23 (2000), e215–e220.

[28] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8–13 2014, Montreal, Quebec, Canada*, Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger (Eds.). 2672–2680. https://proceedings.neurips.cc/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html

[29] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1412.6572

[30] Josif Grabocka, Nicolas Schilling, Martin Wistuba, and Lars Schmidt-Thieme. 2014. Learning time-series shapelets. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 – 27, 2014*, Sofus A. Macskassy, Claudia Perlich, Jure Leskovec, Wei Wang, and Rayid Ghani (Eds.). ACM, 392–401. https://doi.org/10.1145/2623330.2623613

[31] Riccardo Guidotti. 2021. Evaluating local explanation methods on ground truth. *Artificial Intelligence* 291 (2021), 103428.

[32] Riccardo Guidotti and Anna Monreale. 2020. Data-agnostic local neighborhood generation. In *20th IEEE International Conference on Data Mining, ICDM 2020, Sorrento, Italy, November 17–20, 2020*, Claudia Plant, Haixun Wang, Alfredo Cuzzocrea, Carlo Zaniolo, and Xindong Wu (Eds.). IEEE, 1040–1045. https://doi.org/10.1109/ICDM50108.2020.00122

[33] Riccardo Guidotti, Anna Monreale, Fosca Giannotti, Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. 2019. Factual and counterfactual explanations for black box decision making. *IEEE Intell. Syst.* 34, 6 (2019), 14–23. https://doi.org/10.1109/MIS.2019.2957223

[34] Riccardo Guidotti, Anna Monreale, Stan Matwin, and Dino Pedreschi. 2019. Black box explanation by learning image exemplars in the latent feature space. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 11906)*, Ulf Brefeld, Élisa Fromont, Andreas Hotho, Arno J. Knobbe, Marloes H. Maathuis, and Céline Robardet (Eds.). Springer, 189–205. https://doi.org/10.1007/978-3-030-46150-8_12

[35] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. 2018. Local rule-based explanations of black box decision systems. *CoRR* abs/1805.10820 (2018). arXiv:1805.10820 http://arxiv.org/abs/1805.10820

[36] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. 2019. A survey of methods for explaining black box models. *ACM Computing Surveys (CSUR)* 51, 5 (2019), 93.

[37] Riccardo Guidotti, Anna Monreale, Francesco Spinnato, Dino Pedreschi, and Fosca Giannotti. 2020. Explaining any time series classifier. In *2nd IEEE International Conference on Cognitive Machine Intelligence, CogMI 2020, Atlanta, GA, USA, October 28–31, 2020*. IEEE, 167–176. https://doi.org/10.1109/CogMI50398.2020.00029

[38] Maël Guillemé, Véronique Masson, Laurence Rozé, and Alexandre Termier. 2019. Agnostic local explanation for time series classification. In *31st IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2019, Portland, OR, USA, November 4–6, 2019*. IEEE, 432–439. https://doi.org/10.1109/ICTAI.2019.00067

[39] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27–30, 2016*. IEEE Computer Society, 770–778. https://doi.org/10.1109/CVPR.2016.90

[40] Geoffrey E. Hinton and Ruslan R. Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science* 313, 5786 (2006), 504–507.

[41] Lu Hou, James T. Kwok, and Jacek M. Zurada. 2016. Efficient learning of timeseries shapelets. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12–17, 2016, Phoenix, Arizona, USA*, Dale Schuurmans and Michael P. Wellman (Eds.). AAAI Press, 1209–1215. https://doi.org/10.1609/aaai.v30i1.10178

[42] Cun Ji, Chao Zhao, Shijun Liu, Chenglei Yang, Li Pan, Lei Wu, and Xiangxu Meng. 2019. A fast shapelet selection algorithm for time series classification. *Computer Networks* 148 (2019), 231–240. https://doi.org/10.1016/j.comnet.2018.11.031

[43] Shalmali Joshi, Oluwasanmi Koyejo, Warut Vijitbenjaronk, Been Kim, and Joydeep Ghosh. 2019. Towards Realistic Individual Recourse and Actionable Explanations in Black-Box Decision Making Systems. arXiv:1907.09615 [cs.LG]

[44] Isak Karlsson, Panagiotis Papapetrou, and Henrik Boström. 2016. Generalized random shapelet forests. *Data Mining and Knowledge Discovery* 30, 5 (2016), 1053–1085.

[45] Isak Karlsson, Jonathan Rebane, Panagiotis Papapetrou, and Aristides Gionis. 2020. Locally and globally explainable time series tweaking. *Knowledge and Information Systems* 62, 5 (2020), 1671–1700.

[46] Eamonn J. Keogh and Michael J. Pazzani. 2000. Scaling up dynamic time warping for datamining applications. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Boston, Massachusetts, USA) *(KDD '00)*. Association for Computing Machinery, New York, NY, USA, 285–289. https://doi.org/10.1145/347090.347153

[47] Eamonn J. Keogh and Thanawin Rakthanmanon. 2013. Fast shapelets: A scalable algorithm for discovering time series shapelets. In *Proceedings of the 13th SIAM International Conference on Data Mining, May 2–4, 2013. Austin, Texas, USA*. SIAM, 668–676. https://doi.org/10.1137/1.9781611972832.74

[48] Eamonn J. Keogh, Li Wei, Xiaopeng Xi, Stefano Lonardi, Jin Shieh, and Scott Sirowy. 2006. Intelligent icons: Integrating lite-weight data mining and visualization into GUI operating systems. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), 18-22 December 2006, Hong Kong, China*. IEEE Computer Society, 912–916. https://doi.org/10.1109/ICDM.2006.90

[49] Rebecca Killick, Paul Fearnhead, and I. A. Eckley. 2012. Optimal detection of changepoints with a linear computational cost. *J. Amer. Statist. Assoc.* 107 (12 2012), 1590–1598. https://doi.org/10.1080/01621459.2012.737745

[50] Been Kim, Oluwasanmi Koyejo, and Rajiv Khanna. 2016. Examples are not enough, learn to criticize! Criticism for Interpretability. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5–10, 2016, Barcelona, Spain*, Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett (Eds.). 2280–2288. https://proceedings.neurips.cc/paper/2016/hash/5680522b8e2bb01943234bce7bf84534-Abstract.html

[51] Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1312.6114

[52] Tamara G. Kolda and Brett W. Bader. 2009. Tensor decompositions and applications. *SIAM Review* 51, 3 (2009), 455–500.

[53] Himabindu Lakkaraju, Stephen H. Bach, and Jure Leskovec. 2016. Interpretable decision sets: A joint framework for description and prediction. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13–17, 2016*, Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi (Eds.). ACM, 1675–1684. https://doi.org/10.1145/2939672.2939874

[54] Orestis Lampridis, Laura State, Riccardo Guidotti, and Salvatore Ruggieri. 2022. Explaining short text classification with diverse synthetic exemplars and counter-exemplars. *Machine Learning* (2022), 1–34.

[55] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki. 2018. Comparison-based inverse classification for interpretability in machine learning. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems. Theory and Foundations*, Jesús Medina, Manuel Ojeda-Aciego, José Luis Verdegay, David A. Pelta, Inma P. Cabrera, Bernadette Bouchon-Meunier, and Ronald R. Yager (Eds.). Springer International Publishing, Cham, 100–111.

[56] Thach Le Nguyen, Severin Gsponer, and Georgiana Ifrim. 2017. Time series classification by sequence learning in all-subsequence space. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE).* IEEE, 947–958.

[57] Thach Le Nguyen, Severin Gsponer, Iulia Ilie, Martin O'Reilly, and Georgiana Ifrim. 2019. Interpretable time series classification using linear models and multi-resolution multi-domain symbolic representations. *Data Mining and Knowledge Discovery* 33, 4 (2019), 1183–1222.

[58] Ritchie Lee, Mykel J. Kochenderfer, Ole J. Mengshoel, and Joshua Silbermann. 2018. Interpretable categorization of heterogeneous time series data. In *Proceedings of the 2018 SIAM International Conference on Data Mining, SDM 2018, May 3–5, 2018, San Diego Marriott Mission Valley, San Diego, CA, USA*, Martin Ester and Dino Pedreschi (Eds.). SIAM, 216–224. https://doi.org/10.1137/1.9781611975321.25

[59] Yen-Hsien Lee, Chih-Ping Wei, Tsang-Hsiang Cheng, and Ching-Ting Yang. 2012. Nearest-neighbor-based approach to time-series classification. *Decision Support Systems* 53, 1 (2012), 207–217.

[60] Guiling Li, Shaolin Xu, Senzhang Wang, and S. Yu Philip. 2023. Forest based on Interval Transformation (FIT): A time series classifier with adaptive features. *Expert Systems with Applications* 213 (2023), 118923.

[61] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. 2007. Experiencing SAX: A novel symbolic representation of time series. *Data Mining and Knowledge Discovery* 15, 2 (2007), 107–144.

[62] Jason Lines, Luke M. Davis, Jon Hills, and Anthony J. Bagnall. 2012. A shapelet transform for time series classification. In *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12–16, 2012*, Qiang Yang, Deepak Agarwal, and Jian Pei (Eds.). ACM, 289–297. https://doi.org/10.1145/2339530.2339579

[63] Jason Lines, Sarah Taylor, and Anthony J. Bagnall. 2018. Time series classification with HIVE-COTE: The hierarchical vote collective of transformation-based ensembles. *ACM Trans. Knowl. Discov. Data* 12, 5 (2018), 52:1–52:35. https://doi.org/10.1145/3182382

[64] Scott M. Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4–9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 4765–4774. https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html

[65] Qianli Ma, Wanqing Zhuang, Sen Li, Desen Huang, and Garrison Cottrell. 2020. Adversarial dynamic shapelet networks. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 04 (Apr. 2020), 5069–5076. https://doi.org/10.1609/aaai.v34i04.5948

[66] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian J. Goodfellow. 2015. Adversarial autoencoders. *CoRR* abs/1511.05644 (2015). arXiv:1511.05644 http://arxiv.org/abs/1511.05644

[67] Matthew Middlehurst, James Large, and Anthony J. Bagnall. 2020. The canonical interval forest (CIF) classifier for time series classification. In *2020 IEEE International Conference on Big Data (IEEE BigData 2020), Atlanta, GA, USA, December 10–13, 2020*, Xintao Wu, Chris Jermaine, Li Xiong, Xiaohua Hu, Olivera Kotevska, Siyuan Lu, Weija Xu, Srinivas Aluru, Chengxiang Zhai, Eyhab Al-Masri, Zhiyuan Chen, and Jeff Saltz (Eds.). IEEE, 188–195. https://doi.org/10.1109/BigData50022.2020.9378424

[68] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence* 267 (2019), 1–38.

[69] Yao Ming, Huamin Qu, and Enrico Bertini. 2018. RuleMatrix: Visualizing and understanding classifiers with rules. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2018), 342–352.

[70] Saumitra Mishra, Bob L. Sturm, and Simon Dixon. 2017. Local interpretable model-agnostic explanations for music content analysis. In *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23–27, 2017*, Sally Jo Cunningham, Zhiyao Duan, Xiao Hu, and Douglas Turnbull (Eds.). 537–543. https://ismir2017.smcnus.org/wp-content/uploads/2017/10/216_Paper.pdf

[71] Felix Mujkanovic, Vanja Doskoc, Martin Schirneck, Patrick Schäfer, and Tobias Friedrich. 2020. timeXplain - A framework for explaining the predictions of time series classifiers. *CoRR* abs/2007.07606 (2020). arXiv:2007.07606 https://arxiv.org/abs/2007.07606

[72] Meinard Müller. 2007. Dynamic time warping. *Information Retrieval for Music and Motion* (2007), 69–84.

[73] R. Olszewski, R. Maxion, and D. Siewiorek. 2001. Generalized feature extraction for structural pattern recognition in time-series data. Carnegie Mellon University.

[74] Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. 2020. Learning model-agnostic counterfactual explanations for tabular data. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20–24, 2020*, Yennun Huang, Irwin King, Tie-Yan Liu, and Maarten van Steen (Eds.). ACM / IW3C2, 3126–3132. https://doi.org/10.1145/3366423.3380087

[75] Dino Pedreschi, Fosca Giannotti, Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, and Franco Turini. 2019. Meaningful explanations of black box AI decision systems. In *The Thirty-third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-first Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth*

*AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 9780–9784. https://doi.org/10.1609/aaai.v33i01.33019780

[76] Vitali Petsiuk, Abir Das, and Kate Saenko. 2018. RISE: Randomized input sampling for explanation of black-box models. In *British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3–6, 2018*. BMVA Press, 151. http://bmvc2018.org/contents/papers/1064.pdf

[77] C. Ratanamahatana and Eamonn J. Keogh. 2005. Three myths about dynamic time warping data mining. In *SDM*, Hillol Kargupta, Jaideep Srivastava, Chandrika Kamath, and Arnold Goodman (Eds.). SIAM, 506–510. https://doi.org/10.1137/1.9781611972757.50

[78] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13–17, 2016*, Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi (Eds.). ACM, 1135–1144. https://doi.org/10.1145/2939672.2939778

[79] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-precision model-agnostic explanations. In *Proceedings of the Thirty-second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th Innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2–7, 2018*, Sheila A. McIlraith and Kilian Q. Weinberger (Eds.). AAAI Press, 1527–1535. https://doi.org/10.1609/aaai.v32i1.11491

[80] Alejandro Pasos Ruiz, Michael Flynn, James Large, Matthew Middlehurst, and Anthony Bagnall. 2021. The great multivariate time series classification bake off: A review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery* 35, 2 (2021), 401–449.

[81] Patrick Schäfer. 2015. The boss is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery* 29 (2015), 1505–1530.

[82] Patrick Schäfer and Ulf Leser. 2017. Multivariate time series classification with WEASEL+MUSE. *CoRR* abs/1711.11343 (2017). arXiv:1711.11343 http://arxiv.org/abs/1711.11343

[83] Pavel Senin and Sergey Malinchik. 2013. SAX-VSM: Interpretable time series classification using SAX and vector space model. In *2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, December 7–10, 2013*, Hui Xiong, George Karypis, Bhavani Thuraisingham, Diane J. Cook, and Xindong Wu (Eds.). IEEE Computer Society, 1175–1180. https://doi.org/10.1109/ICDM.2013.52

[84] Mattia Setzu, Riccardo Guidotti, Anna Monreale, Franco Turini, Dino Pedreschi, and Fosca Giannotti. 2021. GLocalX - From local to global explanations of black box AI models. *Artif. Intell.* 294 (2021), 103457. https://doi.org/10.1016/j.artint.2021.103457

[85] Pang-Ning Tan, Michael S. Steinbach, Anuj Karpatne, and Vipin Kumar. 2019. *Introduction to Data Mining (Second Edition)*. Pearson. https://www-users.cse.umn.edu/%7Ekumar001/dmbook/index.php

[86] Ninad Thakoor and Jean Gao. 2005. Shape classifier based on generalized probabilistic descent method with hidden Markov descriptor. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, Vol. 1. IEEE, 495–502.

[87] Andreas Theissler, Francesco Spinnato, Udo Schlegel, and Riccardo Guidotti. 2022. Explainable AI for time series classification: A review, taxonomy and research directions. *IEEE Access* 10 (2022), 100700–100724. https://doi.org/10.1109/ACCESS.2022.3207765

[88] Jeffrey Scott Vitter. 1984. Faster methods for random sampling. *Commun. ACM* 27, 7 (1984), 703–718.

[89] Sandra Wachter, Brent Mittelstadt, and Luciano Floridi. 2017. Why a right to explanation of automated decision-making does not exist in the general data protection regulation. *International Data Privacy Law* 7, 2 (2017), 76–99.

[90] Jun Wang, Arvind Balasubramanian, Luis Mojica de La Vega, Jordan R. Green, Ashok Samal, and Balakrishnan Prabhakaran. 2013. Word recognition from continuous articulatory movement time-series data using symbolic representations. In *Proceedings of the Fourth Workshop on Speech and Language Processing for Assistive Technologies, SLPAT 2013, Grenoble, France, August 21–22, 2013*, Jan Alexandersson, Peter Ljunglöf, Kathleen F. McCoy, François Portet, Brian Roark, Frank Rudzicz, and Michel Vacher (Eds.). Association for Computational Linguistics, 119–127. https://aclanthology.org/W13-3919/

[91] Yichang Wang, Rémi Emonet, Élisa Fromont, Simon Malinowski, and Romain Tavenard. 2020. Adversarial regularization for explainable-by-design time series classification. In *32nd IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2020, Baltimore, MD, USA, November 9-11, 2020*. IEEE, 1079–1087. https://doi.org/10.1109/ICTAI50040.2020.00165

[92] Zhendong Wang, Isak Samsten, Rami Mochaourab, and Panagiotis Papapetrou. 2021. Learning time series counterfactuals via latent space representations. In *Discovery Science - 24th International Conference, DS 2021, Halifax, NS, Canada, October 11–13, 2021, Proceedings (Lecture Notes in Computer Science, Vol. 12986)*, Carlos Soares and Luís Torgo (Eds.). Springer, 369–384. https://doi.org/10.1007/978-3-030-88942-5_29

[93] Tom White. 2016. Sampling generative networks: Notes on a few effective techniques. *CoRR* abs/1609.04468 (2016). arXiv:1609.04468 http://arxiv.org/abs/1609.04468

[94] Mathias Wilhelm, Daniel Krakowczyk, Frank Trollmann, and Sahin Albayrak. 2015. eRing: Multiple finger gesture recognition with one ring using an electric field. In *Proceedings of the 2nd international Workshop on Sensor-based Activity Recognition and Interaction, iWOAR 2015, Rostock, Germany, June 25–26, 2015*, Bodo Urban and Thomas Kirste (Eds.). ACM, 7:1–7:6. https://doi.org/10.1145/2790044.2790047

[95] Lexiang Ye and Eamonn J. Keogh. 2009. Time series shapelets: A new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 – July 1, 2009*, John F. Elder IV, Françoise Fogelman-Soulié, Peter A. Flach, and Mohammed Javeed Zaki (Eds.). ACM, 947–956. https://doi.org/10.1145/1557019.1557122

[96] Sung Whan Yoon, Jun Seo, and Jaekyun Moon. 2019. TapNet: Neural network augmented with task-adaptive projection for few-shot learning. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9–15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 7115–7123. http://proceedings.mlr.press/v97/yoon19a.html

[97] Shichao Zhang and Jiaye Li. 2023. KNN classification with one-step computation. *IEEE Trans. Knowl. Data Eng.* 35, 3 (2023), 2711–2723. https://doi.org/10.1109/TKDE.2021.3119140

[98] Shichao Zhang, Jiaye Li, and Yangding Li. 2023. Reachable distance function for KNN classification. *IEEE Trans. Knowl. Data Eng.* 35, 7 (2023), 7382–7396. https://doi.org/10.1109/TKDE.2022.3185149

Received 28 July 2022; revised 7 August 2023; accepted 6 September 2023