Explaining Any Time Series Classifier

Riccardo Guidotti*†, Anna Monreale*, Francesco Spinnato*, Dino Pedreschi* and Fosca Giannotti*
* University of Pisa, Italy, {name.surname}@unipi.it
† ISTI-CNR Pisa, Italy, {name.surname}@isti.cnr.it

2020 IEEE Second International Conference on Cognitive Machine Intelligence (CogMI) | 978-1-7281-4144-2/20/$31.00 ©2020 IEEE | DOI: 10.1109/COGMI50398.2020.00029

***Abstract*—We present a method to explain the decisions of black box models for time series classification. The explanation consists of factual and counterfactual shapelet-based rules revealing the reasons for the classification, and of a set of exemplars and counter-exemplars highlighting similarities and differences with the time series under analysis. The proposed method first generates exemplar and counter-exemplar time series in the latent feature space and learns a local latent decision tree classifier. Then, it selects and decodes those respecting the decision rules explaining the decision. Finally, it learns on them a shapelet-tree that reveals the parts of the time series that must, and must not, be contained for getting the returned outcome from the black box. A wide experimentation shows that the proposed method provides faithful, meaningful and interpretable explanations.**

***Index Terms*—Explainable AI; Time Series Classification; Shapelet-based Rules; Exemplars and Counter-Exemplars;**

I. Introduction

The increasing availability of data stored in the form of time series such as electrocardiogram records, motion sensors data, climate measurements, stock indices, etc., contributed to the diffusion of a wide range of time series classifiers [1] employed on a broad array of applications: from the automated detection of heart diseases to the identification of stock market anomalies. There are various distinct groups of time series classifiers: similarity-based and interval-based [2], dictionary-based [3], shapelet-based [4], [5], [6], [7], and deep leaning-based [8], [9]. The advent of deep architectures such as Residual [10] and Convolutional [11] Neural Networks has brought significant advantages in terms of accuracy and resistance to noise. Indeed, the best time series classifiers, are proved to be deep neural networks or ensemble-based classifiers using a hierarchical structure with probabilistic voting [9].

The drawback of neural networks or complex ensembles lies in the inherent opaqueness of this kind of classifiers, often referred to as "black box" models [12], due to the hidden internal structure and complex decision process which is not human understandable [13]. However, in high-stakes decision making, such as clinical diagnosis, the *explanation aspect* of automated time series classifiers adopted by AI systems becomes the crucial building brick of a trustworthy interaction between the human expert and the machine. Meaningful explanations [14] of time series classification would augment the cognitive ability of human experts, such as medical doctors, to make informed and accurate decisions, better support their 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* [4]. Shapelet decision trees [4] and shapelet transforms [5] extract the shapelets from the time series of the training set by selecting the sub-sequences with high discriminatory power, and exploit them for the classification process. In [7] an approach is proposed that extracts shapelet trees by solving an optimization problem. Other interpretable shapelet-based classifiers [15], [16] can manage multivariate/different length time series. Alternative approaches for mining discriminatory subsequences are the Matrix Profile [17] and SAX approximation [18], [19]. Unfortunately, all such methods are heavily behind black-box time series classifiers in terms of accuracy and stability, especially in presence of noisy data [9].

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 method unveiling the logic of *any* black box classifiers operating on time series. In particular, LASTS is a local model-agnostic shapelet-based explanation method. Given a specific time series x labelled with class y by the black-box, LASTS returns as explanation a factual and counterfactual *shapelet-based rule* that shows the reasons for the classification of x in terms of logic conditions, indicating the sub-sequences that must, and must not, be contained in x to obtain the outcome y returned by the black box. In addition, the explanation consists of a set of *exemplar* and *counter-exemplar* time series. Exemplars are instances classified with the same label of the specific time series being explained and highlight similarities and common parts responsible for the classification. On the other hand, counter-exemplars are instances similar to the one explained but with a different label, and provide evidence of how the time series should be "morphed" for being classified with a different label.

In line with recent studies on explanation [20], [21], we tackle the time series black box outcome explanation problem by deriving a *local* explanation from the understanding of the behavior of the black box in the *neighborhood* of the instance to explain [22]. Inspired by [23], we overcome state-of-the-art limitations by exploiting *(i)* autoencoders [24] for generating, encoding and decoding the local neighborhood, *(ii)* a latent decision tree for selecting exemplars and counter-exemplars [25], and *(iii)* a shapelet tree [7] for designing an explanation which is meaningful, useful and easy to understand.

We emphasize that the *counterfactual* components of LASTS's explanation, i.e., counter-exemplars and counterfactual rules, are becoming an essential ingredient in explainable AI methods [26], [27], [28]. While factual, direct explanations such as decision rules [29], and features importance [20], [21], are crucial for understanding the reasons for a certain outcome,

978-1-7281-4144-2/20/$31.00 ©2020 IEEE
DOI 10.1109/CogMI50398.2020.00029

Authorized licensed use limited to: University of Pisa. Downloaded on December 11,2025 at 11:58:33 UTC from IEEE Xplore. Restrictions apply.

a counterfactual reveals what should change in a given instance to obtain a different classification outcome [26]. The importance of counterfactuals is that they help people in reasoning on the cause-effect relations between observed features and classification outcome. For instance, [23] proposes counter-exemplar images highlighting the similarities and differences between same-class and other-class instances. Our objective is to provide the user with an explanation equipped with factual and counterfactual rules, and with exemplar and counter-exemplar time series, offering an in-depth understanding of the local decision of the black box.

We present a wide experimentation on four datasets of univariate time series and three black box classifiers. We empirically demonstrate that LASTS overtakes existing explanation methods [30], [20], [21] based on feature importance (re-adapted for time series) providing faithful, stable, useful and really understandable explanations.

The rest of the paper is organized as follows. Section II discusses related works. Section III formalizes the problem faced and introduces basic concepts for the explanation method, which is described in Section IV. Section V presents the experiments. Finally, Section VI summarizes our contribution, its limitations, and future research directions.

II. Related Work

Research on black box explanation has recently received much attention [31], [32], [22]. Such an increasing interest is driven by the idea of adopting into AI systems opaque classifiers accompanied by explanation methods such that high performance and explainability can coexist. Explanation methods can be categorized with respect to two aspects. Model-specific *vs* model-agnostic, depending on whether the explanation method exploits knowledge of the internal structure of the black box or not. Local *vs* global, depending on whether the explanation is provided for any specific instance or for the logic of the black box as a whole. The proposed explanation method LASTS extends a line of research on *local*, *model-agnostic* methods originated with [20] and extended in different directions by [33] and by [25], [23].

LIME and SHAP are two of the most well known model and data agnostic local explanation methods. LIME [20] 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 analyzed. The explanation consists of the feature importance of the linear model. The user-determined number of features required by LIME is a clear limitation of the method. SHAP [21] 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 for each feature the unique additive importance.

Besides being model-agnostic, LIME and SHAP are also theoretically not tied to a specific type of data. Indeed in [30], LIME and SHAP are employed for time series explanation after an a-priori segmentation of the time series that can cause a loss of information. On the other hand, the proposed explanation method LASTS is specifically designed for time series and does not require any a-priori segmentation or discretization.

Besides, we advance the state-of-the-art by providing rich explanations made of shapelet-based rules that highlight the classification reasons, and exemplars/counter-exemplars providing further evidence for the observed outcome.

III. Setting The Stage

We address the *black box outcome explanation problem* [22] in the domain of time series classification. A classification dataset X, Y consists of a set $X = \{x_1, x_2, \ldots, x_n\} \in \mathbb{R}^{n \times m}$ of univariate time series with l labels (or classes) assigned to each time series in the vector $Y \in \mathbb{N}^n$. A time series x consists of m time points $x = \{t_1, t_2, \ldots, t_m\} \in \mathbb{R}^m$. For instance, a time series can model an EEG and different labels can indicate the presence/absence of an epileptic seizure. Given a not interpretable, i.e., black box, time series classifier b and a time series x classified by b, i.e., $b(x) = y$, our aim is to provide an explanation e for the decision $b(x) = y$. We use the notation $b(X) = Y$ as a shorthand for $\{b(x) \mid x \in X\} = Y$. We assume that b can be queried at will. More formally, we aim to address the following problem:

Definition 3.1: Let b be a not interpretable time series classifier, and x a time series whose decision $y = b(x)$ 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.*

In order to keep our paper self-contained, we summarize in the following key definitions and concepts necessary to comprehend the proposed explanation method.

A. Autoencoders

In an explanation process it is crucial to respect the distributions of real data in synthetically generated examples created for studying the black box behavior in the neighborhood of the instance to explain. We ensure this property by using autoencoders (AE) [24]. An AE is a type of neural network trained for learning a representation that reduces the dimensionality from m to k and captures non-linear relationships. An *encoder* $\zeta : \mathbb{R}^m \rightarrow \mathbb{R}^k$, and a decoder *decoder* $\eta : \mathbb{R}^k \rightarrow \mathbb{R}^m$ are simultaneously learnt with the objective of minimizing the *reconstruction loss*. Starting from the encoding $z = \zeta(x)$, the autoencoder tries to reconstruct a representation as close as possible to its original input $x \simeq \tilde{x} = \eta(z)$. After the learning, the *decoder* can be used to reconstruct instances never observed, aiming to be used as a generative model. In the literature, there exist several variants designed to guarantee useful properties on the learnt representations such as Generative Adversarial Network (GAN) [34], Variational Autoencoders (VAE), and Adversarial Autoencoders (AAE) [35].

B. Local Rule-based Explanation

Explainable AI methods are increasingly using counterfactuals for helping people to trust explanations by ensuring they identify cause-effect relations between events [27]. Indeed, the local rule-based explanation returned by LORE [25] consists of

Authorized licensed use limited to: University of Pisa. Downloaded on December 11,2025 at 11:58:33 UTC from IEEE Xplore. Restrictions apply.

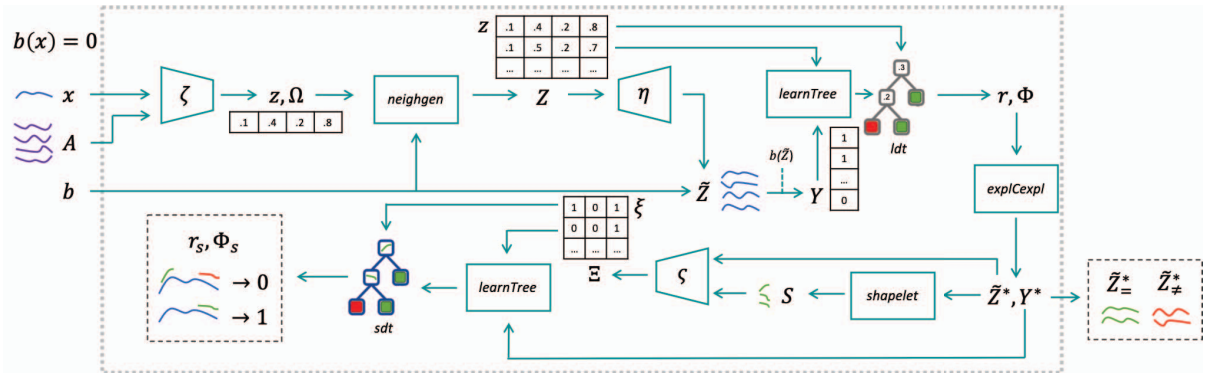

Fig. 1. LASTS architecture. LASTS takes as input the time series x, the black box b and some knowledge on time series A. It uses the AE ζ and η for generating Z and for selecting exemplars and counter-exemplars $\tilde{Z}^*$. From $\tilde{Z}^*$ it extracts the shapelets S and retrieves the shapelet-based rules r_s, Φ_s. The output explanation $e = \langle r_s, \Phi_s, \tilde{Z}^* \rangle$ is contained in the black dashed rectangles.

(i) a *factual* rule r, corresponding to the path in the surrogate tree that explains why x has been labeled as y by b, and *(ii)* a set of *counterfactual* rules Φ, explaining which changes in x would invert the class y assigned by b. LORE learns on the synthetic neighborhood Z of x an interpretable local decision tree that reproduces the behavior of b. The neighborhood Z is generated through a *genetic* algorithm to account for instances similar to x and with the same label, i.e., $y = b(x)$, and for instances similar to x but with a different label, i.e., $y \neq b(x)$. The local explanation including factual and counterfactual rules is derived from the structure of the tree.

C. Time-Series Shapelets and Shapelet Trees

Providing meaningful, useful and usable explanations is fundamental for an explanation method [14], [32]. We try to satisfy these requirements by exploiting time series shapelets. The scientific literature has shown that methods based on shapelets, besides being fast [6], [36], can also be interpretable [4]. Thus, we employ shapelet discovery to capture the reasons for the classification of a time series x with a certain label, i.e., $y = b(x)$. Shapelets are discriminative subsequences of time series that best predict the target class value [7]. A shapelet $s = \{t_1, \ldots, t_{m'}\}$ of length $m' < m$ is an ordered sequence of values. We indicate with S the h-most informative shapelets with respect to a dataset X, Y, and we define the distance between a time series series x_i and a shapelet s_j as the minimum distance $w_{i,j}$ among the distances between the shapelet s_j and each sub-sequence of x_i with length m' [4], [37]. In turn, each distance value is the average squared difference between each point in the shapelet s_j and the aligned point in the sub-sequence of x_i.

The method proposed in [7] learns an optimal set of shapelets S without exploring all possible candidates. Every shapelet in S can have a different length, and the number of shapelts h is obtained using an heuristic [7]. The procedure starts with rough initial guesses for the shapelets in S, and then iteratively optimizes the shapelets by minimizing a classification loss function. In particular, shapelets are updated in a stochastic gradient descent optimization fashion, by taking steps towards maximal prediction accuracy. The minimum distances to the shapelets S can be used to transform [5] the time series $X \in \mathbb{R}^{n \times m}$ into a new representation $\hat{\Xi} \in \mathbb{R}^{n \times h}$. In other words, a time series x_i can be represented with a vector $\hat{\xi}_i = \{v_1, \ldots, v_h\} \in \mathbb{R}^h$ where each value $v_j \in \hat{\xi}_i$ is the minimum distance of x with the shapelet $s_j \in S$. Hence, existing classification methods [7] can be trained on $\hat{\Xi}, Y$.

IV. Explaining Time Series Classifiers

LASTS is a Local Agnostic Subsequence-based Time Series explainer solving the black box outcome explanation problem. Given a black box b and a time series x, the human-interpretable explanation $e \in E$ returned by LASTS for the classification $y = b(x)$ is composed of *(i)* a *shapelet-based rule* and a set of *shapelet-based counterfactual rules*, *(ii)* a set of *exemplars* and *counter-exemplars*. The shapelet-based rule shows the shapelets contained (and not contained) in x responsible for the class y, vice-versa, the counterfactuals highlight how x should have been changed in order to have a different class value. Exemplar and counter-exemplar time series illustrate instances 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. Therefore, the explanations returned by LASTS satisfy the requirements of counterfactuability, usability, and meaningfulness [27], [14], [32].

Besides, the black box b and the time series x to explain, LASTS requires as input: *(i)* an encoder ζ and decoder η for modeling times series in a simplified representation, *(ii)* a set of known time series A for the neighborhood generation process. The explanation process described in Algorithm 1 and in Figure 1 involves the following steps. First, LASTS generates a neighborhood in the latent feature space exploiting the AE. Then, it learns a latent decision tree on that latent neighborhood providing local decision and counter-factual rules. Such rules are used to identify exemplars and counter-exemplars and the corresponding reconstructed time series are used to learn shapelets. Finally, a shapelet-based decision

Authorized licensed use limited to: University of Pisa. Downloaded on December 11,2025 at 11:58:33 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1: LASTS(x, b, ζ, η, A)

Input : x - time series, b - black box, ζ - encoder, η - decoder, A - known time series
Output: e - explanation

1 $z \leftarrow \zeta(x)$; $\Omega \leftarrow \zeta(A)$; // encode data into the latent space
2 $Z \leftarrow neighgen(z, b, \zeta, \eta, \Omega)$; // generate latent neighborhood
3 $\tilde{Z} \leftarrow \eta(Z)$; $Y \leftarrow b(\tilde{Z})$; // decode and classify neighborhood
4 $ldt \leftarrow learnTree(Z, Y)$; // learn latent decision tree
5 $r, \Phi \leftarrow extractRules(z, ldt)$; // extract latent rules
6 $\tilde{Z}^*, Y^* \leftarrow exCex(r, \Phi, Z, \tilde{Z}, Y)$; // sel (counter-)exemplars
7 $S \leftarrow shapelets(\tilde{Z}^*, Y^*)$; // learn shapelet
8 $\xi \leftarrow \varsigma(\tilde{z}, S)$; $\Xi \leftarrow \varsigma(\tilde{Z}^*, S)$; // data as shapelet presence
9 $sdt \leftarrow learnTree(\Xi, Y^*)$; // learn shapelet-based tree
10 $r_s, \Phi_s \leftarrow extractRules(\xi, sdt)$; // extract shapelet rules
11 **return** $e = \langle r_s, \Phi_s, \tilde{Z}^* \rangle$; // return explanation

tree is learned on the reconstructed exemplars and counter-exemplars and used to retrieve the shapelet-based factual rule and counterfactual rules completing the explanation. Details of each step are presented in the following. The explanation of the time series x in Figure 2 is used as running example.

A. Latent Encoding

The time series $x \in \mathbb{R}^m$ is passed to the encoder $z = \zeta(x)$ of a trained AE (line 1). It returns the latent representation $z \in \mathbb{R}^k$ using k latent features with $k \ll m$. k is kept low to avoid possible high dimensionality problems. To capture sequential, nonlinear, and temporal relationships between different time points of the time series, we need AEs based on *convolutional* layers for both the encoder ζ and decoder η (used in lines 1-3). Although traditionally developed for two-dimensional image data, convolutional neural networks can be used to model univariate time series [9]. Inspired by [9], we model the structure of the encoder ζ and decoder η as illustrated in Figure 3. A convolutional layer operates over a one-dimensional sequence, i.e., a time series, and it is followed by other convolutional layers. We use a total of $g + 1$ layers. The last convolutional layer aggregates the output of previous layers. Finally, a dense fully connected layer distills the m features extracted into the k latent space. A flatten layer transposes the convolutional layer as input for the dense layer. The *convolution* basically applies a sliding filter or kernel with length d over a time series x. The filter can be seen as a generic nonlinear transformation of x. The result of a convolution can be considered as a univariate time series v that passes through a filtering process. If u filters are applied the result of a layer is a multivariate time series $V \in \mathbb{R}^{m \times u}$ whose dimensions are equal to the number u of filters used. For each layer we use an increasing (decreasing) number of filters $u{=}2^i{\in}[1, g]$, and decreasing (increasing) kernel sizes d. The time series x of Figure 2 with length $m{=}128$ is represented with a latent space with $k = 2$ in the blue cross in Figure 5. Similarly to [23], the idea of using a latent representation, and therefore of an AE, is to boost the neighborhood generation process

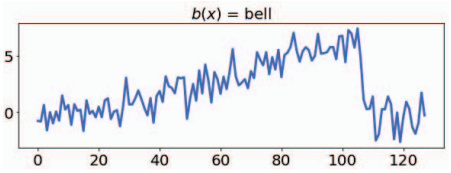

Fig. 2. Time series x classified as $b(x) = bell$.

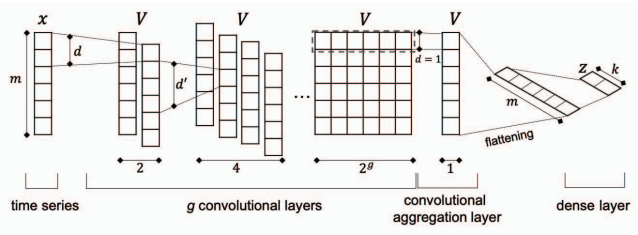

Fig. 3. Structure of the convolutional encoder ζ mapping the times series $x \in \mathbb{R}^m$ into the latent space $z \in \mathbb{R}^k$. The decoder has the opposite structure.

such that variations to few latent features can generate time series which are both realistic and synthetic and depends of a limited number of latent features that can be controlled.

B. Neighborhood Generation

Exploiting z, b and the set of known time series A, LASTS generates a set Z of N instances in the latent feature space with characteristics close to those of z. The neighborhood generation $neighgen$ (line 2) can be implemented adopting different strategies ranging from a pure random approach like in LIME [20] to using a given distribution and a genetic algorithm maximizing a fitness function like in LORE [25]. LASTS uses the *genetic* approach that guarantees a higher density around the instance to explain z. The genetic neighborhood generation $neighgen$ requires *(i)* the distribution of the latent features that is extracted from $\Omega = \zeta(A)$ (line 1) to perform the random mutations, and *(ii)* the black box b for querying b with the synthetic reconstructed instances. As a consequence, $neighgen$ also uses the decoder η. The goal of $neighgen$ is to find instances similar to z with different class values such that we can learn on Z a simple predictor able to locally simulate the behavior of b. Hence, Z is composed of instances with different classes, i.e., $Z = Z_{=} \cup Z_{\neq}$ such that $b(\eta(z')) = b(x)$ for instances $z' \in Z_{=}$, and $b(\eta(z')) \neq b(x)$ for $z' \in Z_{\neq}$. Since $neighgen$ exploits a knowledge A of time series for generating $\tilde{Z} = \eta(Z)$ (line 3) then, time series in $\tilde{Z}$ are *admissible* by design, i.e., they are obtained from latent instances taking values from a valid domain because their generation happens in a domain with distributions drawn from $\Omega = \zeta(A)$. The fact that the autoencoder is fundamental to generate meaningful time series is shown in Section V.

Figure 5 *(left)* shows the latent neighborhood Z highlighting in green the instances $Z_{=}$ labelled as *bell*, and in red the instances $Z_{\neq}$ labelled with a different class value. We observe how the latent space perfectly summarizes the separation of the different class values unveiling a decision boundary easy

Authorized licensed use limited to: University of Pisa. Downloaded on December 11,2025 at 11:58:33 UTC from IEEE Xplore. Restrictions apply.

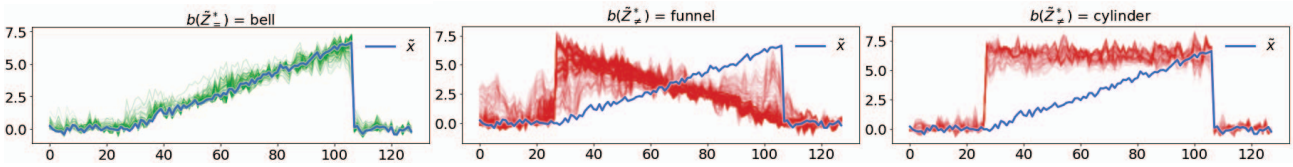

Fig. 4. Exemplars $\tilde{Z}^*$ (in green) and counter-exemplars $\tilde{Z}^*_=$ (in red) respecting the latent rules r, Φ, respectively.

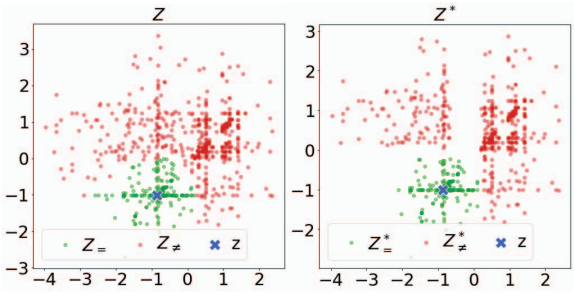

Fig. 5. *Left:* latent neighborhood Z with class equals to x in green $Z_=$, with different to x in red $Z_\neq$. *Right:* latent (counter-)exemplars Z respecting r, Φ.

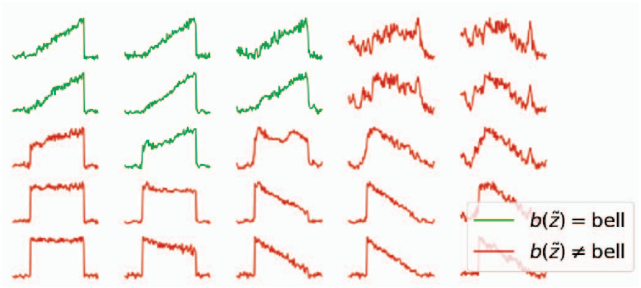

Fig. 6. Exemplar/counter-exemplar morphing matrix showing the time series shapes and class changes (green same class, red different class).

to detect with a simple classifier. Moreover, both $Z_=$ and $Z_\neq$ forming the neighborhood surround in a dense way x, yet helping to capture the black box behavior "locally" around x.

C. Local Latent Rules and Exemplars

Given the local latent neighborhood Z and $Y = b(\tilde{Z})$ (line 3), LASTS builds a latent decision tree to locally mimic the behavior of b (line 4). Querying ldt with z allows to extract the latent decision rule r and counter-factual rules Φ (line 5). The premise p of the rule $r{=}p{\rightarrow}y$ is the conjunction of the splitting conditions in the tree that is satisfied by z. For Φ, LASTS selects q as the conjunction of splitting conditions labeling an instance with $\hat{y} \neq y$ and minimizing the number of falsified splitting conditions w.r.t. p. In our example $r = \{-2.15{<}z_0{\leq}0.10 \wedge z_1{\leq}-0.22\} \rightarrow$ *bell*, $\Phi = \{\{z_0{>}0.18\} \rightarrow$ *funnel*, $\{z_0{\leq}-0.67 \wedge 0.12{<}z_1{\leq}2.92\} \rightarrow$ *cylinder*$\}$.

Exemplars and counter-exemplars are identified by the $exCex$ module (line 6). Indeed, $exCex$ selects from $Z_=$ the latent instances respecting r, namely $Z^*_=$, and from $Z_\neq$ the latent instances respecting one of the counterfactual rules in Φ, namely $Z^*_\neq$. The reconstructed time series in $\tilde{Z}$ corresponding to the instances in $Z^*_=$ and $Z^*_\neq$ form the exemplars $\tilde{Z}^*_=$ and $\tilde{Z}^*_\neq$ counter-exemplars $\tilde{Z}^*_\neq$. Figure 5 *(right)* shows the latent neighborhood Z^* containing only the instances respecting r or Φ. Selecting only these instances the local decision boundary becomes very clear. Figure 4 illustrates the exemplars (in green) and counter-exemplars (in red). Exemplars have a slope very similar to x and with very low noise. *funnel* counter-exemplars can have either a completely different slope with low noise or a slope similar to the one of a *bell* with very high noise. The few *cylinder* counter-exemplars have a very different trend. The exemplars and counter-exemplars $\tilde{Z}^*$ are a powerful prototype-based part of the explanation e that allows us to understand when a time series morphs from a class to another one. In Figure 6 we horizontally vary the latent feature z_0 and vertically vary feature z_1. In line with the previous observations, increasing values of z_0 increases the noise that leads to a change of slope (1^{st} and 2^{nd} rows) towards the class *funnel*. Increasing values of z_1 rises the left part of the time series morphing the class into *cylinder* (1^{st} and 2^{nd} columns).

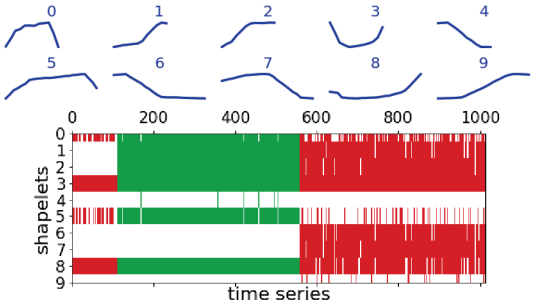

Fig. 7. *Top:* Set of shapelets S. *Bottom:* Shapelet matrix Ξ. Each column represents a time series. For each row, if the shapelet is not contained, the cell is white; it is green if it is an exemplar, red if it is a counter-exemplar.

D. Shapelets Mining and Rules Extraction

Given the exemplars and counter-exemplars $\tilde{Z}^*$ and the labels Y^*, (line 7) LASTS extracts the shapelets S using the $shapelet$ module [7] that learns the h most discriminative shapelets S with respect to Y^*. Figure 7 (left) shows the shapelets S of our example. As an alternative, we plan to implement the $shapelet$ module using the Matrix Profile [17], [38] or SAX [18], [19] for shapelet discovery. On top of S, LASTS performs the shapelet transformation ς encoding a time series into a space of presence/absence of shapelets (line 8). ς encodes in the matrix $\hat{\Xi} \in \mathbb{R}^{n \times h}$ the minimum distances of the

Authorized licensed use limited to: University of Pisa. Downloaded on December 11,2025 at 11:58:33 UTC from IEEE Xplore. Restrictions apply.

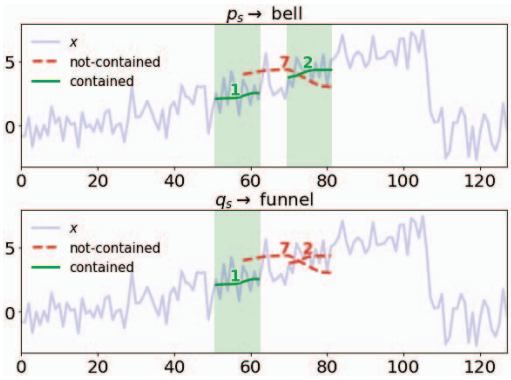

Fig. 8. *Top:* factual rule. *Bottom:* counterfactual rule. All the shapelets in the rules are shown at their best alignment.

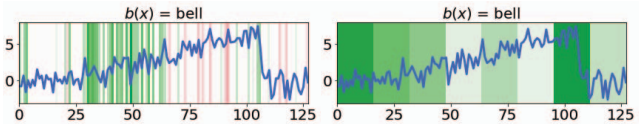

Fig. 9. *Left:* SHAPP explanation. *Right:* SHAPS explanation. `cbf` dataset. The darker the color the higher the importance.

shapelet $s_j \in S$ from the time series $\tilde{z}_i \in \tilde{Z}^*$. Then, it analyzes the distances in $\hat{\Xi}$, sets a threshold τ, and returns Ξ such that $\Xi_{i,j} = 1$, *if* $\hat{\Xi}_{i,j} \leq \tau$, and 0 *otherwise*. The matrix Ξ for our running example is reported in Figure 7 (bottom). Each column represents a time series in $\tilde{Z}^*$. For each row, a colored cell indicates the presence of the corresponding shapelet, while a white cell indicates the shapelet absence. Exemplars contain shapelets 0, 1, 2, 3, 5, 8, *funnel* counter-exemplars are similar but without 0, 5 and with 6, 7; *cylinder* counter-exemplars only contains shapelets 0, 3, 5, 8.

Given Ξ and Y^*, LASTS trains a shapelet-based decision tree classifier sdt that allows to identify shapelet-based (counter-)factual rules r_S, Φ_S (lines 9 - 10). LASTS adopts decision trees because *(i)* decision rules can be naturally derived from a root-leaf path in a decision tree; and, *(ii)* counter-factual rules can be extracted by symbolic reasoning [25]. In our example we have: $r_s = \{s_1 \in x \wedge s_2 \in x \wedge s_7 \notin x\} \rightarrow$ *bell*, $\Phi = \{\{s_1 \in x \wedge s_2 \notin x \wedge s_7 \notin x\} \rightarrow$ *funnel*$\}$. The corresponding rules are shown in Figure 8. The visual rules illustrate the position of the shapelets[1] that must be contained and which are those that must not be contained (at their best alignment with x). Looking at the rules, a user can truly understand the reasons for the classification and how the time series should have been for having another outcome. LASTS adopts decision trees in both the latent and shapelet space for two different reasons: *(i)* the latent decision tree filters exemplars and counter-exemplars clearing the local decision boundary, *(ii)* the shapelet-based decision tree is used to retrieve the shapelet-based factual and counter factual explanation rules.

[1]Alignments can be preformed only moving the shapelets along the x-axis.

TABLE I
DATASETS DESCRIPTION AND BLACK BOX MODELS ACCURACY.

dataset	n	m	l	$\lvert X_{bb}\rvert$	$\lvert X_{ae}\rvert$	$\lvert X_e\rvert$	RES	CNN	KNN
`cbf`	600	128	3	268	115	36	1.00	1.00	1.00
`esr`	4,600	178	2	2,060	706	50	.978	.972	.865
`har`	10,299	561	6	4,116	1,411	50	.952	.891	.901
`poc`	2,658	80	2	1,008	345	50	.829	.768	.783

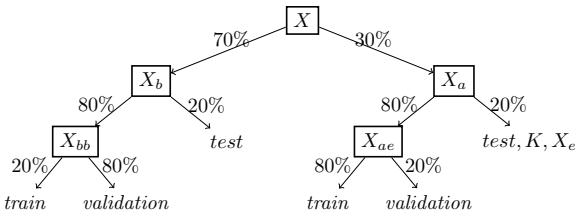

Fig. 10. Datasets partitioning.

V. EXPERIMENTS

In this section we show the faithfulness, stability, and meaningfulness of LASTS explanations[2].

We experimented LASTS on four *datasets*: *cylinder-bell-funnel* [39] (`cbf`, used in our example) has time series with three classes having different shapes, *epileptic seizure recognition* [40] (`esr`) is composed of EEG records with two classes indicating the presence/absence of an epileptic seizure, *human activity recognition* [41] (`har`) [41] contains signals recorded from a smartphone while performing six different activities, *phalanges outlines correct* [42] (`poc`) contains time series outlining hand bones of the images published in [43]. We name X_{bb} the partition of the dataset used for training the black boxes, X_{ae} the partition used for training the autoencoders, and X_e the partition from which we selected the instances to explain in the experiments. Datasets details are in Table I. We partitioned each dataset X as illustrated in Figure 10. 70% of X, namely X_b, was dedicated to the black boxes, the remaining 30%, namely X_a, to the explainers. We used 80% of X_b, called X_{bb}, for training the black boxes and the 20%, called for testing the black boxes. In turn, 80% of X_{bb} is used for training and 20% for validation. On the other hand, we used 80% of X_a for training the autoencoders, namely X_{ae}, 80% of which was used for the training and the remaining 20% for validation. The remaining 20% of X_a was used for *(i)* testing the autoencoders, *(ii)* providing knowledge A to the explainers, and *(iii)* selecting a set X_e of instance to explain.

We trained and explained the following *black box classifiers*: RES, CNN, KNN[3]. *ResNet* (RES), introduced in [10], was implemented in `keras` according to [44]. It is composed of three residual blocks each one containing three convolutional layers, a global average pooling layer, and a dense layer. In particular, the layers inside each residual block have respectively $64, 128, 256$ filters, of size $8, 5, 3$. We trained RES with a batch size of 16, monitoring the loss, with a patience

[2]Code available at: https://github.com/fspinna/TS_AgnosticLocalExplainer
[3]https://bit.ly/2uEEMke, https://keras.io/, https://scikit-learn.org/.

Authorized licensed use limited to: University of Pisa. Downloaded on December 11,2025 at 11:58:33 UTC from IEEE Xplore. Restrictions apply.

TABLE II
AEs RECONSTRUCTION ERROR IN TERMS OF MSE.

dataset	AE	VAE	AAE	AEL
cbf	1.050	1.050	1.260	6.810
esr	1e+04	3e+04	2e+04	6e+04
har	0.009	0.024	0.008	0.071
poc	0.001	0.022	0.001	0.008

TABLE III
CLASSIFICATION ACCURACY FOR RECONSTRUCTED INSTANCES.

dataset	black box	AE	VAE	AAE	AEL
esr	RES	.955	.941	.923	.511
	CNN	.964	.959	.937	.502
	KNN	.959	.801	.869	.602
har	RES	.950	.776	.946	.337
	CNN	.910	.771	.912	.170
	KNN	.968	.769	.955	.394
poc	RES	.972	.639	.926	.676
	CNN	.972	.500	.981	.769
	KNN	.963	.648	.963	.778

parameter of 50 epochs. As optimizer we selected Adam, with the default `keras` parameters: $learning_rate = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, minimizing the sparse categorical crossentropy. The *Convolutional Neural Network* [9] (CNN) was implemented in `keras` and it is composed of three convolutional layers with ReLu activation function, each one followed by batch normalization and a dropout layer, and a final average global pooling layer followed by a dense layer to output the class[4]. The dropout rate is 0.3. The remaining parameters are the same as for RES. Finally, we tested *K-Nearest Neighbor* [45] (KNN) as implemented by `scikit-learn`. The k value was selected via grid search using the `scikit-learn` package, searching from a value $k \in [1, |X_{bb}|]$. In the grid search we also tested two kind of weighted distances: Euclidean and Manhattan. The best results were obtained using the euclidean distance, while the best k are 1 for `cbf` and `esr`, 4 for `har` and `poc`. Classification performance are reported in Table I. RES has higher and more stable performance across the various datasets.

We experimented standard (AE), variational (VAE) and adversarial autoencoders (AAE) with the structure illustrated in Figure 3 and an AE using one LSTM [9] layer (AEL). AE, VAE and AAE have $g = 8$ layers for the encoder ζ and $g = 8$ layers[5] for the decoder η. As activation functions: ***ReLu*** for `har` and ***Elu*** for `cbf`, `esr` and `poc`. The autoencoders built for `poc` have a maxpooling layer instead of a dense layer before the latent space. This choice, together with the need of a large latent space for `poc` if compared with the other datasets, is due to the the difficulty of reconstruction that is in turn probably connected to the different nature of the dataset that is obtained as transformations from image processing. We trained all the autoencoders for 2000 epochs, with a batch size equals to 16. As optimizer we selected 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 exception of the AAE and VAE. The AAEs discriminator is a network with two layers of 100 units each with ***ReLU*** activation functions, Adam optimizer, and minimizes the MSE for the autoencoder, and the binary crossentropy for the discriminator. The autoencoder is trained with default `keras` parameters, while the discriminator with $learning_rate = 0.0002$, $\beta_1 = 0.5$, $\beta_2 = 0.999$. The loss weight are to 0.999 and 0.001. In the VAE the loss functions are the MSE and the Kullback–Leibler loss. Finally, for the LSTM autoencoder we used ***LeakyReLu*** activation function with $\alpha = 0.1$ for all the datasets. We measured the performance of the autoencoders by means of the reconstruction error between the original and reconstructed time series in terms of *Mean Squared Error* (MSE, the lower the better), and in terms of *accuracy* of the classifiers on the reconstructed time series (the higher the better). We selected the dimension k of latent space by starting with $k = 2$ and iteratively building autoencoders with an increasing value of k until the accuracy for the reconstructed instances increases and reaches 0.9, but also keeping $k \leq m/2$. This process[6] lead to the usage of the following values for the number of latent features k for the various datasets: `cbf` $k = 2$, `esr` $k = 30$, `har` $k = 50$, `poc` $k = 40$. In Tables II and III we report the MSE and the accuracy on reconstructed time series[7]. We observe that the AE has on average the best performance across the various datasets and classifiers. As consequence, we selected AE as autoencoder for LASTS and we used the corresponding encoder ζ and decoder η.

For LASTS we adopted the following hyper-parameters. The neighborhood generation $neighgen$ is run with neighborhood size equals to $N = 1000$ latent instances, 10 generations, the *normalized Euclidean* distance was used for the genetic fitness function, probability of mutation equals to 0.5, probability of crossover equals to 0.7. The shapelet extraction module $shapelet$ is implemented using the `tslearn` python package[8]. For the shapelet learning we selected the *Stochastic Gradient Descent* optimizer with a training of 50 epochs implemented according to [7]. We selected the number h and length of the shapelets using the the heuristic proposed in [7] which takes as parameters the size of the data set, the length of the time series, the number of classes, and α, the fraction of the length of time series to be used for base shapelet length, and β, the number of different shapelet lengths to use. As suggested in [7], we used α=0.1 and β=2. We transformed the distances $\hat{\Xi}$ into shapelet presence/absence Ξ choosing a threshold distance τ. We adopted a grid-search strategy with the hyper-parameters of the `sklearn` decision tree[9].

[4]In particular the layers have respectively $16, 32, 64$ filters, of size $8, 5, 3$.

[5]The encoder has $u \in \{2, 4, 8, 16, 32, 64, 128, 256\}$ filters of size $d \in \{21, 18, 15, 13, 11, 8, 5, 3\}$ per layer, respectively. The decoder has a symmetric structure with the parameters in the reverse order.

[6]We also plan to test the dimensionality estimation proposed in [46].

[7]`cbf` is missing from Table III because nearly all the accuracy are one.

[8]https://bit.ly/2GDV6o1, https://bit.ly/2U7ERaU

[9]τ tested for the deciles of the distribution of all the distances in $\hat{\Xi}$ from 0.1 to 0.9 with 0.1 step-length, $min_samples_split \in \{0.002, 0.01, 0.05, 0.1, 0.2\}$, $min_samples_leaf \in \{0.001, 0.01, 0.05, 0.1, 0.2\}$, $max_depth \in \{None, 2, 4, 6, 8, 10, 12, 16\}$.

Authorized licensed use limited to: University of Pisa. Downloaded on December 11,2025 at 11:58:33 UTC from IEEE Xplore. Restrictions apply.

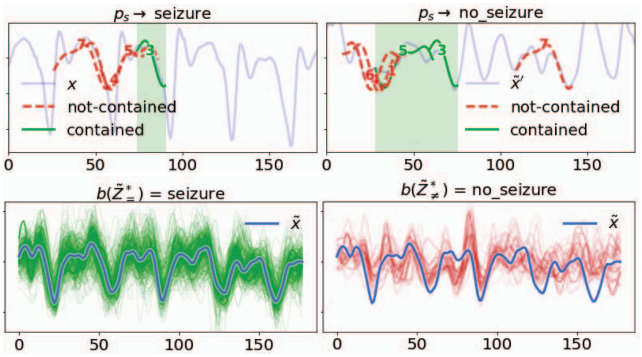

Fig. 11. *1st:* shapelet rule. *2nd:* shapelet counterfactual applied to counter-exemplar. *3rd:* exemplars. *4th:* counter-exemplars. esr dataset. b = RES.

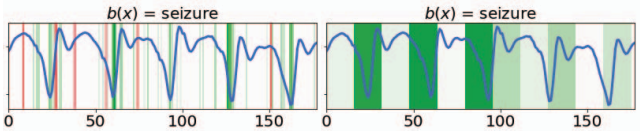

Fig. 12. *Left:* SHAPP. *Right:* SHAPS. esr dataset. b = RES.

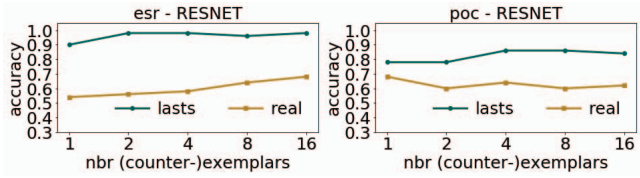

Fig. 13. 1-NN accuracy varying n (counter-)exemplars.

A. Comparing Time Series Explanations

Before assessing quantitatively the effectiveness of LASTS, we compare LASTS explanations with those returned by SHAP [21] employed to solve the time series black box outcome explanation problem as in [30]. In particular, we instantiate SHAP as: SHAPP every time point in the time series becomes a feature and SHAP is run as for tabular data, and SHAPS an a-priori segmentation with predefined length is performed on the time series and a linear interpolation (similar to image explanation) is used as base value by SHAP.

In Figure 9 we show the explanations of SHAPP and SHAPS. The punctual explanation of SHAPP (top) is similar to a saliency maps highlighting pixels, and therefore it is very difficult to understand. It seems that time points between 30 and 60 are important for recognizing the class *bell*. The explanation returned by SHAPS (top) highlights various segments and puts more importance on the first and penultimate sequence not considering that also a *cylinder* can respect this explanation rising in the second or third segment.

Besides the visual utility demonstrated by exemplars and counter-exemplars in Figure 4, it is evident that the shapelet-based rules of LASTS clarifies better than the explanations of SHAP the reasons for the *bell* classification. The factual rule visualized in Figure 8 tells that a *bell* time series must have an increasing central part (s_1, s_2) between 50 and 80, and not have an increasing part (s_7) in the same interval. Indeed, the shapelet-based counterfactual shows that if s_2 is missing the black box recognize the time series as a *funnel*.

Figures 11 and 12 allow to compare the explanations obtained using LASTS, SHAPP and SHAPS for an instance of esr. Observing the exemplars, counter-exemplars, shapelet-based rule, and a counter-exemplar respecting a counterfactual rule in Figure 11 we can grasp the reasons for which x (in blue in the plots) has been labelled as *seizure*. The rule shows that the cause lies in the presence of s_3 with a sharp decrease of the EEG signal and, how highlighted by the counterfactual, by the absence of s_5 that would have meant the existence of a large bell generally representing healthy subjects. The statement above is confirmed by the exemplars with many sharp fluctuations and by the counterfactuals, which, on the contrary, have more relaxed and irregular EEG signals. From the explanations in Figure 12, we understand that the reasons for the class *seizure* are mainly related to the lowest values in the EEG according to SHAPP, or to the valleys according to SHAPS. However, also time series with *no seizure* have low values and valleys. This is why SHAP-based explanation can be not entirely meaningful or useful.

B. Are (Counter-)Exemplars Useful?

Since we cannot validate the usefulness of exemplars and counter-exemplars with an experiment involving humans, inspired by [47], we tested their effectiveness replacing humans with a memory-based machine learning technique, i.e., a k-NN classifier. This experiment gives an objective and indirect estimation of the usefulness of exemplars and counter-exemplars. For each instance $x \in X_e$, we randomly select n exemplars and counter-exemplars, one each for each class. Then, we use the selected (counter-)exemplars to train a 1-NN and we classify x. We compare this approach with a 1-NN trained on n *real* time series per class randomly selected from $X_e - \{x\}$. As distance function we use in both cases the Euclidean distance.

The average accuracy of the two approaches named LASTS and *real* is reported in Figure 13 observed varying n for each $x \in X_e$. We notice that LASTS clearly overcomes *real*. Indeed, exemplars and counter-exemplars help in discovering the decision boundary and in highlighting similarities and differences. The performance of LASTS are on average high and constant in every datasets revealing the also few exemplars and counter-exemplars are a good proxy for recognizing the classification outcome. On the other hand, the accuracy of *real* only increases for esr with increasing n. This result shows that not every time series is suitable for recognizing a class, but they are as such only if carefully selected like LASTS does.

C. Are Surrogates Faithful to the Black Box?

We evaluate the *faithfulness* [12], [22] of the surrogates adopted by LASTS by measuring the ability of the shapelet-based decision tree sdt of mimicking the behavior of the black box b. We measure the fidelity comparing $Y = b(X_e)$ and $Y' = \{y' | \forall x \in X_e, y' = sdt_x(\varsigma(\eta(\zeta(x))))\}$ in terms

Authorized licensed use limited to: University of Pisa. Downloaded on December 11,2025 at 11:58:33 UTC from IEEE Xplore. Restrictions apply.

TABLE IV
FIDELITY. HIGHEST VALUES ARE UNDERLINED.

dataset	black box	LASTS	RLASTS	SBGDT
cbf	RES	1.00	1.00	1.00
	CNN	.889	.917	1.00
	KNN	1.00	1.00	1.00
esr	RES	1.00	.500	.960
	CNN	1.00	.500	.960
	KNN	.940	.820	.920
har	RES	.980	.460	.420
	CNN	.860	.440	.400
	KNN	.980	.400	.380
poc	RES	.500	.400	.660
	CNN	.960	.400	.660
	KNN	.940	.440	.740

TABLE V
MEDIAN INCOHERENCE. LOWEST VALUES ARE UNDERLINED.

	RES			CNN			KNN		
	LASTS	SHAPP	SHAPS	LASTS	SHAPP	SHAPS	LASTS	SHAPP	SHAPS
cbf	.16	.99	.94	.52	.99	.94	.03	.99	.94
esr	1.0	.99	.92	.81	.99	.98	.87	.99	.94
har	.68	.99	.99	.77	1.0	1.0	.23	1.0	.99
poc	.90	.99	.98	.85	.99	.97	.58	.99	.99

of *accuracy* where sdt_x is the local shapelet-based decision tree learned for x. We compare LASTS against RLASTS that, similarly to LIME, uses a *random* neighborhood generation. In addition, we show that extracting an explanation from the neighborhood of a given instance is a winning strategy compared to an approach that builds a single *global* interpretable classifier. Thus, we compare LASTS with a shapled-based global decision tree classifier (SBGDT) trained on the partition X_{ae} used by LASTS for training the autoencoders. In this case the fidelity is calculated as the accuracy between $Y = b(X_e)$ and $Y' = sbgdt(X_e)$.

Table IV reports the values of the fidelity. We observe that LASTS outperforms both RLASTS and SBGDT. Indeed, the genetic approach allows to better explore the neighborhood than a random one, while the global approach fails for datasets like har and poc in discovering the local discriminative boundaries. The non-parametric Friedman test compares the average ranks of explanation methods over multiple datasets and black boxes w.r.t. the fidelity. The null hypothesis that all methods are equivalent is rejected ($p-value < 0.01$).

D. Are the Explanations Coherent?

Explainers *stability* is a fundamental property to gain the user trust and to guarantee a reliable service [48]. We asses the stability of LASTS in terms of *coherence*: *similar* time series labelled with the same class should get *similar* explanations [23]. Inspired by the local Lipschitz estimation [49] we design the *incoherence* of an explanation e for a time series x as: $\mathcal{I}_{\mathcal{N}_x}(x) = sim(e_{x^{(f)}}, e_x)/sim(e_{x^{(c)}}, e_x)$ where $\mathcal{N}_x = \{x_i \in X_e \mid dist(x_i, x) \leq \epsilon\}$ is the neighborhood of x defined in terms of a distance function $dist$, $x^{(c)}$ and $x^{(f)}$ are respectively the closest and furthest time series to x in $\mathcal{N}_x$ in terms of $dist$, $e_x, e_{x^{(c)}}, e_{x^{(f)}}$ are their explanations, and

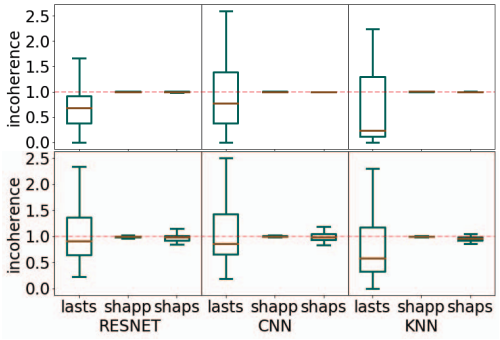

Fig. 14. Boxplots of incoherence for har and poc.

sim is a similarity function between explanations[10]. Thus, the lower is the incoherence $\mathcal{I}$, the more stable is the explanation. Values lower than one indicates that the similarity between the explanations of similar instances is higher than the similarity between dissimilar instances. We compare the stability of LASTS with SHAPP and SHAPS[11]. The similarity sim in the incoherence formula is calculated as $1/(1-d)$. For LASTS d is the average distance between couple of shapelets contained and not contained in the shapelet-based rule[12]. For SHAP d is the difference in absolute values between shap values[13].

Table V reports the median incoherence. Results show that LASTS returns explanations much more coherent than those returned by SHAP and with values markedly lower. The non-parametric Friedman test for the incoherence rejected the null hypothesis that all methods are equivalent ($p-value < 0.0001$). Figure 14 reports the box plots showing all the incoherence values for the har dataset. We notice that incoherence of LASTS explanations can vary a lot, while the incoherence of SHAP explanations is stably around one. This fact is not necessarily a weakness because indicates that also among similar time series LASTS finds a variegate array of similar explanations but with different causes.

VI. CONCLUSION

We have presented LASTS, a local model-agnostic subsequence-based explainer exploiting shapelets for black box classifiers working on time series. LASTS succeeds in addressing the time series black box outcome explanation

[10]For $\mathcal{N}_x$, instead of setting a radius ϵ, we required that $|\mathcal{N}_x| \leq k$ with $k = 30$. Thus, $x^{(f)}$ is the k-th closest instance to x.

[11]For SHAPS we consider each time series with segments having length 16.

[12]Due to cases in which a rule has only shapelets contained or not contained.

[13]As distance $dist$ we used the Euclidean distance. For both LASTS and SHAP we calculated the similarity as $1/(1-d)$ where d is the distance between two explanations. We used two different distances d because of the different nature of the explanations returned by LASTS and SHAP. For LASTS $d(e^{(1)}, e^{(2)})$ is the average distance between the Euclidean distances among the couple of shapelets in the rules $r^{(1)}$ $r^{(2)}$ which are contained in the respective time series $x^{(1)}$ and $x^{(2)}$, and among those which are not contained. If there are not correspondences a default infinite distance is set penalizing the result. For SHAPP and SHAPS $d(e^{(1)}, e^{(2)})$ is the Euclidean distance of the vectors between shap values. We can compare the incoherence for explanations with different nature because the measure is normalized w.r.t. the similarity of the explanations between x and the furthest time series in $\mathcal{N}_x$.

Authorized licensed use limited to: University of Pisa. Downloaded on December 11,2025 at 11:58:33 UTC from IEEE Xplore. Restrictions apply.

problem exploiting *two data representations*: the latent space and the shapelets. The latent space leverages a smart neighborhood generation that permits to observe the black box behavior with similar and different time series. The shapelets allow understanding the logic of the classification showing the reasons for the outcome in terms of subsequences that must, and must not, be contained. An extensive experimentation shows that LASTS outperforms existing explainers in returning meaningful, useful, faithful, and coherent explanations.

The method has some limitations. The shapelet-based rules do not consider multiple alignments of the same shapelet in different points of the time series. Multiple occurrences could help for better explaining a predictive phenomenon. Moreover, LASTS only works for univariate and multi-class time series classifiers. A challenge is to extend LASTS to make it also work for multivariate time series and multilabel [50] classification. Also, technical and conceptual extensions are possible. First, extending LASTS for different types of sequential data like text and shopping transactions. Second, studying the relationship between the latent space and the shapelet space. Third, empowering the expressiveness of the explanations and enabling higher levels of abstraction with grammar-based decision trees [51]. Finally, a human decision-making task driven by LASTS explanations could objectively evaluate the real effectiveness of the explanations.

ACKNOWLEDGMENT

This work is partially supported by the European Community H2020 programme under the funding schemes: INFRAIA-01-2018-2019: Res. Infr. G.A. 871042 *SoBigData++*, G.A. 952026 *Humane AI-Net*, G.A. 825619 *AI4EU*, G.A. 952215 *TAILOR*, and the ERC-2018-ADG G.A. 834756 "XAI: Science and technology for the eXplanation of AI decision making".

REFERENCES

[1] A. Bagnall *et al.*, "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances," *DAMI*, vol. 31, no. 3, pp. 606–660, 2017.
[2] S. F. Boubrahimi *et al.*, "Scalable knn search approximation for time series data," in *ICPR*. IEEE, 2018, pp. 970–975.
[3] M. G. Baydogan, G. Runger, and E. Tuv, "A bag-of-features framework to classify time series," *TPAMI*, vol. 35, no. 11, pp. 2796–2802, 2013.
[4] L. Ye and E. Keogh, "Time series shapelets: a new primitive for data mining," in *SIGKDD*. ACM, 2009, pp. 947–956.
[5] J. Lines, L. M. Davis, J. Hills, and A. Bagnall, "A shapelet transform for time series classification," in *SIGKDD*, 2012, pp. 289–297.
[6] T. Rakthanmanon *et al.*, "Fast shapelets: A scalable algorithm for discovering time series shapelets," in *ICDM*. SIAM, 2013, pp. 668–676.
[7] J. Grabocka, N. Schilling, M. Wistuba, and L. Schmidt-Thieme, "Learning time-series shapelets," in *SIGKDD*. ACM, 2014, pp. 392–401.
[8] Z. Wang *et al.*, "Time series classification from scratch with deep neural networks: A strong baseline," in *IJCNN*. IEEE, 2017, pp. 1578–1585.
[9] H. I. Fawaz *et al.*, "Deep learning for time series classification: a review," *DAMI*, vol. 33, no. 4, pp. 917–963, 2019.
[10] K. He *et al.*, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.
[11] Y. LeCun *et al.*, "Convolutional networks for images, speech, and time series," *HBTNN*, vol. 3361, no. 10, p. 1995, 1995.
[12] A. A. Freitas, "Comprehensible classification models: a position paper," *ACM SIGKDD explorations newsletter*, vol. 15, no. 1, pp. 1–10, 2014.
[13] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint arXiv:1702.08608*, 2017.
[14] D. Pedreschi, F. Giannotti *et al.*, "Meaningful explanations of black box ai decision systems," in *AAAI*, vol. 33, 2019, pp. 9780–9784.
[15] B. Hidasi *et al.*, "Shifttree: an interpretable model-based approach for time series classification," in *ECML-PKDD*. Springer, 2011, pp. 48–64.
[16] E.-Y. Hsu *et al.*, "Multivariate time series early classification with interpretability using deep learning," in *PAKDD*. Springer, 2019.
[17] H. A. Dau *et al.*, "Matrix profile v: A generic technique to incorporate domain knowledge into motif discovery," in *KDD*, 2017, pp. 125–134.
[18] P. Senin *et al.*, "Sax-vsm: Interpretable time series classification using sax and vector space model," in *ICDM*. IEEE, 2013.
[19] L. Nguyen *et al.*, "Interpretable time series classification using linear models and multi-resolution multi-domain symbolic representations," *DAMI*, vol. 33, no. 4, pp. 1183–1222, 2019.
[20] M. T. Ribeiro *et al.*, "Why should i trust you?: Explaining the predictions of any classifier," in *SIGKDD*. ACM, 2016, pp. 1135–1144.
[21] S. M. Lundberg *et al.*, "A unified approach to interpreting model predictions," in *NIPS*, 2017, pp. 4765–4774.
[22] R. Guidotti, A. Monreale, S. Ruggieri *et al.*, "A survey of methods for explaining black box models," *CSUR*, vol. 51, no. 5, p. 93, 2019.
[23] R. Guidotti *et al.*, "Black box explanation by learning image exemplars in the latent feature space," in *ECML-PKDD*. Springer, 2019.
[24] G. E. Hinton *et al.*, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.
[25] R. Guidotti, A. Monreale, F. Giannotti *et al.*, "Factual and counterfactual explanations for black box decision making," *IS*, 2019.
[26] S. Wachter *et al.*, "Why a right to explanation of automated decision-making does not exist in the gdpr," *IDPL*, vol. 7, pp. 76–99, 2017.
[27] R. M. Byrne, "Counterfactuals in explainable artificial intelligence (xai): evidence from human reasoning," in *IJCAI*, 2019, pp. 6276–6282.
[28] A. Apicella *et al.*, "Contrastive explanations to classification systems using sparse dictionaries," in *ICIAP*. Springer, 2019, pp. 207–218.
[29] H. Lakkaraju *et al.*, "Interpretable decision sets: A joint framework for description and prediction," in *SIGKDD*. ACM, 2016, pp. 1675–1684.
[30] U. Schlegel *et al.*, "Towards a rigorous evaluation of xai methods on time series," *arXiv preprint arXiv:1909.07082*, 2019.
[31] A. Adadi *et al.*, "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, 2018.
[32] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artificial Intelligence*, vol. 267, pp. 1–38, 2019.
[33] N. Frosst *et al.*, "Distilling a neural network into a soft decision tree," *arXiv:1711.09784*, 2017.
[34] I. Goodfellow *et al.*, "Generative adversarial nets," in *NIPS*, 2014.
[35] A. Makhzani *et al.*, "Adversarial autoencoders," *1511.05644*, 2015.
[36] M. Linardi *et al.*, "Valmod: A suite for easy and exact detection of variable length motifs in data series," in *ICMD*, 2018, pp. 1757–1760.
[37] L. Ye *et al.*, "Time series shapelets: a novel technique that allows accurate, interpretable and fast classification," *DAMI*, vol. 22, 2011.
[38] C.-C. M. Yeh *et al.*, "Time series joins, motifs, discords and shapelets: a unifying view that exploits the matrix profile," *DAMI*, 2018.
[39] S. Naoki, *Local Feature Extraction and Its Applications*, 2000.
[40] R. Andrzejak *et al.*, "Indications of nonlinear deterministic and finite-dimensional structures in time series of brain activity." *PRE*, 2002.
[41] D. Anguita *et al.*, "A public domain dataset for human activity recognition using smartphones." in *Esann*, vol. 3, 2013, p. 3.
[42] A. J. Bagnall *et al.*, "Predictive modelling of bone age through classification and regression of bone shapes," *CoRR*, vol. 1406.4781, 2014.
[43] F. Cao *et al.*, "Digital hand atlas for web-based bone age assessment: System design and implementation," *CMIG*, vol. 24, pp. 297–307, 2000.
[44] H. I. Fawaz *et al.*, "Data augmentation using synthetic data for time series classification with deep residual networks," *CoRR*, vol. abs/1808.02455, 2018.
[45] J. Bien *et al.*, "Prototype selection for interpretable classification," *The Annals of Applied Statistics*, 2011.
[46] E. Facco *et al.*, "Estimating the intrinsic dimension of datasets by a minimal neighborhood information," *SR*, pp. 1–8, 2017.
[47] B. Kim, "Examples are not enough, learn to criticize!" in *NIPS*, 2016.
[48] R. Guidotti and S. Ruggieri, "On the stability of interpretable models," in *IJCNN*. IEEE, 2019, pp. 1–8.
[49] D. A. Melis and T. Jaakkola, "Towards robust interpretability with self-explaining neural networks," in *NIPS*, 2018.
[50] C. Panigutti *et al.*, "Explaining multi-label black-box classifiers for health applications," in *IWHI*. Springer, 2019, pp. 97–110.
[51] R. Lee *et al.*, "Interpretable categorization of heterogeneous time series data," in *SDM*. SIAM, 2018, pp. 216–224.

Authorized licensed use limited to: University of Pisa. Downloaded on December 11,2025 at 11:58:33 UTC from IEEE Xplore. Restrictions apply.