MIPCE: Generating Multiple Patches Counterfactual-Changing Explanations for Time Series Classification

Hiroyuki Okumura(✉) and Tomoharu Nagao

Yokohama National University, Yokohama, Kanagawa, Japan
okuhiro.8765@gmail.com, nagao@ynu.ac.jp

Abstract. In the development of AI and deep neural networks (DNNs), a growing concern has emerged regarding not only accuracy, but explainability. The corresponding field of research, known as eXplainable AI (XAI), is important because interpreting the predictions of AI helps users make decisions in critical areas such as medicine. XAI has recently gained popularity particularly for counterfactual explanations from a psychological perspective. However, despite recent progress in XAI, few existing methods focus on explaining time series data. We therefore propose Multiple Patches Counterfactual-changing Explanations (MIPCE) for fully convolutional networks (FCNs), which focuses on subsequences of time series, showing the process of change to the counterfactual. First, MIPCE obtains subsequences from features appearing in the FCN, and divides the time series data into patches. Using GPLVM, it then generates the interpretable process of counterfactual change in each patch. We compared our method with other counterfactual methods in terms of proximity, plausibility, and substitutability. These quantitative results indicate that MIPCE outperforms existing methods. In addition, our user test shows that our explanations are useful in helping users understand the decision-making processes of DNNs.

Keywords: XAI · Time Series Classification · Counterfactual Explanations

1 Introduction

Recent years have witnessed a surge of interest in DNNs, particularly in the field of image recognition. Research on time series classification using DNNs has likewise progressed, with FCNs having demonstrated competitive performance, making them promising candidates for real-world applications [10,26]. However, the inner workings of DNNs are a black box, making it difficult for end users to trust model output. To address this problem, researchers have been actively exploring the field of XAI. Methods such as LIME [21], SHAP [19] and CAM [28] have been proposed to provide transparent explanations of model predictions. One approach within XAI is counterfactual explanation, which shows how a

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
L. Iliadis et al. (Eds.): ICANN 2023, LNCS 14259, pp. 231–242, 2023.
https://doi.org/10.1007/978-3-031-44223-0_19

query can be altered to the counterfactual instance in order to change the model's prediction result. Counterfactual explanation not only presents important components of the query that contributed to the prediction, but also suggests the user's next action to change the result. From this perspective, it is said to be psychologically effective [4], with many methods having been proposed [8,13,18]. Although counterfactual explanation has become an increasingly popular XAI field in recent years [20], most existing methods focus on image and tabular data, and few methods have been developed for time series data [7,9,12].

In the field of time series, certain subsequences within the data are considered to have significance [27]. Just as DNNs learn semantic concepts in the image domain [3], they are likely to learn subsequences in time series. To improve end-user understanding and satisfaction, it is effective to present explanations based on these subsequences, in addition to changing the model classification results. Actually, research in the image domain has shown that presenting the meaningful concepts learned by DNNs as explanations has led to increased user satisfaction [1]. Furthermore, in the case of time series data, multiple subsequences contribute to classification [10]. To fully interpret a model's predictions, it is therefore necessary to treat all of these subsequences simultaneously.

Our proposed method, MIPCE, obtains subsequences (referred to as patches) learned by the FCN, and divides the corresponding query into patches. Aside from generating counterfactuals, MIPCE also provides the process of each patch's continuous change to the counterfactual. This is because presenting changes to the counterfactual has been shown to have a positive effect on user understanding and satisfaction [23].

2 Related Works

As mentioned previously, LIME, SHAP, and CAM are widely recognized XAI methods. They provide visual explanations by highlighting the regions that contribute to the classification. In the case of counterfactuals, Watcher proposed a baseline method (W-CF) [25] that generates the counterfactual within a small distance from the query. As extensions of W-CF, many methods in the image domain use generative models, such as generative adversarial networks (GANs), to obtain the counterfactuals [11,17,23]. Furthermore, methods that use features appearing in DNNs along with GANs have also been proposed [13]. These methods are designed to satisfy proximity, which measures the similarity between the query and counterfactual, and plausibility, which determines whether the counterfactual is following the data distribution or is out of distribution (OOD). In addition, counterfactuals must be generated in a form that is recognizable to humans from an XAI perspective [20]. Some methods jointly present the process of change to the counterfactual [11,17]. One such method has demonstrated the effectiveness of its interpretation through expert evaluations [23]. However, it should be noted that these methods do not specifically focus on time series data.

In the field of time series XAI, one popular approach is to focus on time series subsequences known as shapelets [27], and a method has been developed to

explain random forest classification models that are trained with shapelets [12]. Our approach also focus on subsequences, but it differs in terms of the target models and the procedures for obtaining subsequences. Another method is Native-Guide [7], which modifies the results of any classification model by changing part of the query to the nearest-neighbor instance of a different class (denoted as NUN, short for nearest-unlike-neighbor). This method can be applied to DNNs classification models such as FCNs, but there are difficulties in accurately capturing subsequences. It should also be noted that these methods do not have the capability to generate continuous changes to the counterfactual.

3 MultIple Patches Counterfactual-Changing Explanations(MIPCE)

MIPCE divides time series data into subsequences using Gaussian mixture models (GMM), and generates continuous changes from the query to the counterfactual (see Fig. 1). It is necessary for the process of change to follow the principle of proximity in order to provide more interpretable explanations for users. Ideally, continuous changes would gradually approach the counterfactual in the range between the query and counterfactual. In addition, sparsity, defined as the idea of not changing anything except the necessary parts of the query, is also important for interpretability. To generate these ideal explanations, MIPCE uses Gaussian process latent variable models (GPLVM) [14] for each patch.

3.1 Setup and Notation

Assume a two-class FCN classification model [26] (denoted as M) as a black-box model. We represent the input data as $\mathbf{y} \in \mathbb{R}^{T_\mathbf{y}}$, latent variable of the $\mathbf{y}$ as

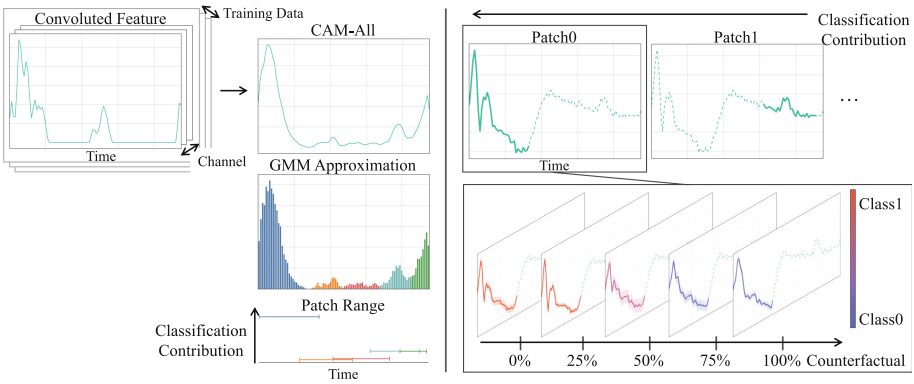


Fig. 1. MIPCE overview. (Left) Time series patch division. (Right) Change to the counterfactual. Green shows the original query, others show changes from the query. (Color figure online)

$\mathbf{z}$, and the feature extracted by the convolution as $\mathbf{X} \in \mathbb{R}^{T_\mathbf{X} \times S}$. S denotes the number of channels in the last convolutional layer. $\{\mathbf{x}_s\}_{s=1}^{S} \geq \mathbf{0}$ by using ReLU activations, and $T_\mathbf{X}$ and $T_\mathbf{y}$ are equal by setting the strides of all convolutional layers to 1. Let R denote the convolutional receptive field. For the query (denoted as q), the classified class by the FCN is represented as c, and the classification probability is represented as $M_c(\mathbf{y})$. For the counterfactual, the classified class and the probability is similarly represented as $c', M_{c'}(\mathbf{y})$. Let $\mathbf{v}_{scaled}$ denote the min-max normalized value for $\mathbf{v}$, and $v_{\arg \text{scaled}_\mathbf{z}}$ denote the scaled v by applying min-max normalization to the set $\mathbb{v}$ ($v \in \mathbb{v}$) obtained by varying $\mathbf{z}$ in its defined range.

3.2 Algorithm

Divide the Time Series Data into Patches (Algorithm 1). Using the features of all N_D training data $\{\mathbf{X}^n\}_{n=1}^{N_D}$, compute a variant of CAM (CAM-All $\in \mathbb{R}^{T_\mathbf{X}}$) that retrieves all features contributing to the classification together:

$$\text{CAM-All} = \Big(\sum_{c \in \{1,2\}} \frac{1}{N_D S} \sum_{n=1}^{N_D} \sum_{s=1}^{S} |w_{s,c}| \mathbf{x}_s^n \Big)_{scaled} \tag{1}$$

$w_{s,c}$ is a weight that connects the s channel's output of the convolution layer to the class c input of the softmax layer in the FCN. The GMM, which uses Dirichlet process [2] (referred to as DPGMM), is then fit to the sampled data points via rejection sampling [5] from the CAM-All. This allows the CAM-All to be divided into clusters in the temporal direction.

Let the minimum, maximum, and mean time steps of each cluster $\Bbbk \in \{\mathbb{1}, \ldots, \mathbb{K}\}$ be denoted as $t_\mathbf{X}^{min_\Bbbk}, t_\mathbf{X}^{max_\Bbbk}$, and $t_\mathbf{X}^{mean_\Bbbk}$ respectively. When $\Bbbk$ is in ascending order, $\Bbbk$ and $\Bbbk + \mathbb{1}$ are merged into a single cluster if:

$$t_\mathbf{X}^{mean_{\Bbbk+\mathbb{1}}} - t_\mathbf{X}^{mean_\Bbbk} \leq R \tag{2}$$

Because $T_\mathbf{X} = T_\mathbf{y}$, clusters in the feature space can be considered as clusters in the input space. Thus, under (2), the representative time step of two clusters in the input space becomes one feature following convolution. Therefore, these two clusters should not be treated independently, as they have a correlation. When we redefine the cluster as $\Bbbk \in \{\mathbb{1}, \ldots, \mathbb{K}\}$, and the time steps as $t_\mathbf{X}^{min_\Bbbk}$ and $t_\mathbf{X}^{max_\Bbbk}$ after the merge process, the range of time steps for patch $\Bbbk$ is:

$$\begin{aligned} \mathbb{T}_\mathbf{y}^\Bbbk &= \{t_\mathbf{y}^{min_\Bbbk}, \ldots, t_\mathbf{y}^{max_\Bbbk}\}, \quad \text{where} \\ t_\mathbf{y}^{min_\Bbbk} &= t_\mathbf{X}^{min_\Bbbk} - \frac{1}{2}R, \quad t_\mathbf{y}^{max_\Bbbk} = t_\mathbf{X}^{max_\Bbbk} + \frac{1}{2}R \end{aligned} \tag{3}$$

Equation (3) calculates the mininum and maximum time steps of input data that will affect to the $\{t_\mathbf{X}^{min_\Bbbk}, \ldots, t_\mathbf{X}^{max_\Bbbk}\}$. Then, the contribution of the patch $\Bbbk$ to the classification is computed via:

$$\text{Contrib}_\Bbbk = \sum_{t \in \mathbb{T}_\mathbf{y}^\Bbbk} \text{CAM-All}_t \tag{4}$$

where CAM-All_t is a t time step value of the CAM-All.

Algorithm 1. Patch Division and Contribution to Classification

Input: $\{\mathbf{X^n}\}_{n=1}^{N_D}$: Convoluted features of all training data, $\mathbf{W}$: Weight matrix with $w_{s,c}$ as its (s, c) element, R: Convolutional receptive field of FCN
1. Compute the CAM-All with (1).
2. Run rejection sampling from the CAM-All.
3. Fit DPGMM to sampled points and obtain $t_{\mathbf{X}}^{min_{\mathbb{k}}}, t_{\mathbf{X}}^{max_{\mathbb{k}}}$ and $t_{\mathbf{X}}^{mean_{\mathbb{k}}}$ of a cluster $\mathbb{k}$.
4. Merge clusters based on (2), and obtain $\mathbb{T}_{\mathbf{y}}^{\mathbb{k}}$ with (3).
5. Compute $\text{Contrib}_{\mathbb{k}}$ of each patch $\mathbb{k}$ with (4).

return: $\mathbb{T}_{\mathbf{y}}^{\mathbb{k}}$ and $\text{Contrib}_{\mathbb{k}}$ of each patch.

Generate Counterfactual Changing (Algorithm 2). When representing latent variables and observational data as $\mathcal{D} = \{(\mathbf{z}_1, \mathbf{y}_1), (\mathbf{z}_2, \mathbf{y}_2), \ldots\}$, the GPLVM's expected value of the predictive distribution for the unknown latent variable $\mathbf{z}^*$ is defined as:

$$\begin{aligned} \mathbb{E}[p(\mathbf{y}^* \mid \mathbf{z}^*, \mathcal{D})] &= \mathbf{k}_*^T \mathbf{K}^{-1} \mathbf{Y} \\ \mathbf{k}_* = (k(\mathbf{z}^*, \mathbf{z}_1), k(\mathbf{z}^*, \mathbf{z}_2), \ldots)^T, \mathbf{Y} &= (\mathbf{y}_1, \mathbf{y}_2, \ldots)^T \end{aligned} \tag{5}$$

k represents the kernel function and $\mathbf{K}$ represents covariance matrix. As GPLVM is commonly used for dimensionality reduction, it is possible to divide the latent space into clusters. Considering three clusters – $\mathbb{c1}$, $\mathbb{c2}$ and $\mathbb{c3}$ – case, (5) can be:

$$\mathbb{E}[p(\mathbf{y}^* \mid \mathbf{z}^*, \mathcal{D})] = (\mathbf{k}_{*,\mathbb{c1}}, \mathbf{k}_{*,\mathbb{c2}}, \mathbf{k}_{*,\mathbb{c3}})\, \mathbf{K}^{-1} (\mathbf{Y}_{\mathbb{c1}}, \mathbf{Y}_{\mathbb{c2}}, \mathbf{Y}_{\mathbb{c3}})^T \tag{6}$$

If we want to obtain $\mathbf{y}^*$ that exists between $\mathbf{Y}_{\mathbb{c1}}$ and $\mathbf{Y}_{\mathbb{c2}}$, this case is difficult to realize due to the influence of $\mathbf{Y}_{\mathbb{c3}}$. The same argument can be applied to the case where we want to obtain the ideal continuous change to the counterfactual. Therefore, it is necessary to select one cluster of each class in advance.

Data Selection. Prepare the query patch $\mathbf{y}_{\mathbb{T}_{\mathbf{y}}^{\mathbb{k}}}^q$ and N_{sim} similar patches of each class c and c' with Euclidean distance. Then, train Bayesian GPLVM [24] with the patches, and apply DPGMM to obtain the latent variable $\mathbf{z}_q$ of the query, class c latent variable clusters $\mathbb{z}_c \in \{\mathbb{1}, \ldots, \mathbb{Z}_c\}$ and so is class c'. When we denote the mean of the $\mathbb{z}_c$ as $\mathbf{z}_{\mathbb{z}_c}$, and the number of elements as $|\mathbb{z}_c|$, score clusters with:

$$\text{Score-}\mathbb{z}_c = \frac{1}{|\mathbb{z}_c|} \sum_{\mathbf{z} \in \mathbb{z}_c} M_c(\mathbb{E}[G_B(\mathbf{z})]) + \alpha_1 (1 - (\|\mathbf{z}_q - \mathbf{z}_{\mathbb{z}_c}\|_2^2)_{\arg\text{scaled}_{\mathbf{z}_{\mathbb{z}_c}}}) + \alpha_2 |\mathbb{z}_c| \tag{7}$$

In (7), Bayesian GPLVM is represented as G_B, and $G_B(\mathbf{z})$ denotes the predictive distribution of $\mathbf{z}$. The first term represents the average patch classification probability of cluster $\mathbb{z}_c$. The second and the third terms are constraints to satisfy proximity and plausibility, as the cluster's elements size reflects the data distribution. Finally, the cluster $\hat{\mathbb{z}}_c$ can be selected with $\arg\max_{\mathbb{z}_c} \{\text{Score-}\mathbb{1}_c, \ldots, \text{Score-}\mathbb{Z}_c\}$. Similarly, we can find $\hat{\mathbb{z}}_{c'}$ for class c'.

We assume a small value of N_{sim}. It is therefore necessary to increase the number of data points in the latent space prior to DPGMM clustering. Bayesian

Algorithm 2. Generate Changes to the Counterfactual Using GPLVM

Input: M: FCN, $\mathbf{y}^q_{\mathbb{T}^{\mathbb{k}}_{\mathbf{y}}}$: The query patch, N_{sim}: The number of similar patches to use

1. Prepare N_{sim} similar patches from class c and c' of $\mathbf{y}^q_{\mathbb{T}^{\mathbb{k}}_{\mathbf{y}}}$.
2. Train Bayesian-GPLVM and obtain latent Gaussian distributions of the query patch, class c and c' patches.
3. Fit DPGMM and obtain $\hat{\mathbb{z}}_c, \hat{\mathbb{z}}_{c'}$ by scoring clusters with (7).
4. Train GPLVM (G) with $\hat{\mathbb{z}}_c, \hat{\mathbb{z}}_{c'}$ patches and explore $(\mathbf{z}_{cf}, \mathbf{z}_{sf})$ with (8) and (9).
5. Obtain $\mathbb{z}_{q\to sf}$ and $\mathbb{z}_{sf\to cf}$ with (10).

return: $G(\mathbb{z}_{q\to sf})$ and $G(\mathbb{z}_{sf\to cf})$

GPLVM is an appropriate choice because it allows sampling from the Gaussian distribution of the latent variable, while having equivalent properties to GPLVM.

Counterfactual Changing. Train GPLVM with the query patch, as well as patches of the clusters $\hat{\mathbb{z}}_c$ and $\hat{\mathbb{z}}_{c'}$. This allows us to obtain the latent variables $\mathbf{z}_q$ of the query, $\{\mathbf{z}^n\}_{n=1}^N$ of the class c and $\{\mathbf{z}^{n'}\}_{n'=1}^{N'}$ of the class c', where $N = |\hat{\mathbb{z}}_c|$ and $N' = |\hat{\mathbb{z}}_{c'}|$. Then, select a $\mathbf{z}_{cf}$ to generate a counterfactual patch:

$$\begin{aligned} \mathbf{z}_{cf} &= \arg\max_{\mathbf{z}^{n'}} \{\text{Score-}\mathbf{z}^{1'}, \ldots, \text{Score-}\mathbf{z}^{N'}\}, \quad \text{where} \\ \text{Score-}\mathbf{z}^{n'} &= M_{c'}(\mathbb{E}[G(\mathbf{z}^{n'})]) + \alpha_3(1 - (\|\mathbf{z}_q - \mathbf{z}^{n'}\|_2^2)_{\text{arg scaled}_{\mathbf{z}^{n'}}}) \end{aligned} \tag{8}$$

In (8), GPLVM is represented as G as well as Bayesian GPLVM in (7). Using $\mathbf{z}_q$ and $\mathbf{z}_{cf}$, explore the latent space $\mathcal{Z}$ to find $\mathbf{z}_{sf}$ to generate a semifactual patch:

$$\begin{aligned} \mathbf{z}_{sf} = \arg\max_{\mathbf{z}\in\mathcal{Z}} &(1 - |0.5 - M_c(\mathbb{E}[G(\mathbf{z})])|_{\text{arg scaled}_{\mathbf{z}}}) \\ &+ \alpha_4(1 - (\|\mathbf{z} - \mathbf{z}_q\|_2^2 + \|\mathbf{z} - \mathbf{z}_{cf}\|_2^2)_{\text{arg scaled}_{\mathbf{z}}}) \end{aligned} \tag{9}$$

The semifactual is the instance when the classification result changes. Equation (9) constrains that $\mathbf{z}_{sf}$ is within the $\mathbf{z}_q$ and $\mathbf{z}_{cf}$ while the classification probability of the patch is 0.5. After acquiring $(\mathbf{z}_q, \mathbf{z}_{sf}, \mathbf{z}_{cf})$, we can obtain the set of internal latent variables $\mathbb{z}_{q\to sf}$ and $\mathbb{z}_{sf\to cf}$ by linearly varying β in (10), where $0 \le \beta \le 1$:

$$\mathbf{z}_{q\to sf} = \beta\mathbf{z}_q + (1-\beta)\mathbf{z}_{sf}, \quad \mathbf{z}_{sf\to cf} = \beta\mathbf{z}_{sf} + (1-\beta)\mathbf{z}_{cf} \tag{10}$$

Then, generate the continuously changing patch from the query to the semifactual and from the semifactual to the counterfactual, by $G(\mathbb{z}_{q\to sf})$ and $G(\mathbb{z}_{sf\to cf})$. Using the linear kernel with an RBF kernel for the GPLVM allows us to generate continuous changes that gradually increase the distance from the query.

The Whole Algorithm. Based on Algorithm 1, divide the query into patches and then generate counterfactual changes in the order of the patches with the highest contribution to the classification using Algorithm 2. By iterating this process until the classification result changes, the final explanation can be obtained.

The end user is presented with the expected value and a 95% confidence interval. During the iterative process, overlapping patches may be used for explanation. In such cases, Algorithm 2 is applied to them as a single patch.

4 Experiments

We verified the effectiveness of MIPCE with five time series datasets from UCR Archive [6]: ECG200, Strawberry, GunPoint, ProximalPhalanxOutlineCorrect (Proximial), and Wafer. Although the Wafer dataset has a test size of 6164, we randomly selected 50 samples from each class in the interest of conserving computational time. In Experiment 1, we compared MIPCE with several existing methods. Experiment 2 was conducted to evaluate continuous changes, whereas Experiment 3 investigated whether users could understand the decision processes of DNNs from explanations.

FCN Settings. The model consists of three convolutional layers with ReLU activations, a global average pooling layer, and a softmax layer. Batch normalization was applied before input to the ReLU. The number of channels in the convolution, and the kernel size, were set in the order of $(128, 256, 128)$ and $(7, 5, 3)$ from the input layer, respectively. This refers to [26] where high accuracy is achieved.

MIPCE Settings. For the GPLVM and Bayesian GPLVM, we set the latent variable dimensions to 2, and used the results of PCA as initial values. Models were trained with Normal $(0, 1)$, Gamma$(3, 1)$ and Gamma$(1, 1)$ as the prior distributions of latent variables, corresponding to parameters of the linear and RBF kernels respectively. Training was conducted over 1000 iterations and optimized with L-BFGS-B [15]. $(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ in the algorithm were all set to 0.1 and $N_{sim} = 15$. We explored the $\mathcal{Z}$ in (9) via grid search, and changed β in (10) so that $\mathbb{z}_{q \to sf}$ and $\mathbb{z}_{sf \to cf}$ were 50 steps each.

4.1 Experiment 1: Counterfactuals

We compared MIPCE with W-CF and Native-Guide in qualitative and quantitative metrics, specifically in terms of proximity, plausibility, and substitutability.

Proximity evaluates the relative distance between the query (q) and counterfactual (CF) by $\frac{d(q,\mathrm{CF})}{d(q,\mathrm{NUN})}$. We employed the L1 norm, L2 norm, and L∞ (L-Inf) norm as d.

Plausibility evaluates whether the counterfactual is OOD with OCSVM [22] and Isolation Forest (IForest) [16]. In addition, we used interpretable metrics called IM1 and IM2, which use an autoencoder [13]. OCSVM and IForest detect OOD based on distance, whereas IM1 and IM2 do so based on features.

Substitutability evaluates whether sufficient classification accuracy is achieved when using counterfactuals as training data [13]. Prepare a k-nearest neighbor classifier k-$\mathrm{NN}_{\mathrm{orig}}$ trained on the original data, and k-$\mathrm{NN}_{\mathrm{CF}}$ trained on the counterfactuals. Then calculate the accuracy in classifying the test dataset and obtain the following ratio: R%-Sub $\equiv \frac{k\text{-}\mathrm{NN}_{\mathrm{CF}}\ \mathrm{Acc.}}{k\text{-}\mathrm{NN}_{\mathrm{orig}}\ \mathrm{Acc.}} \times 100$.

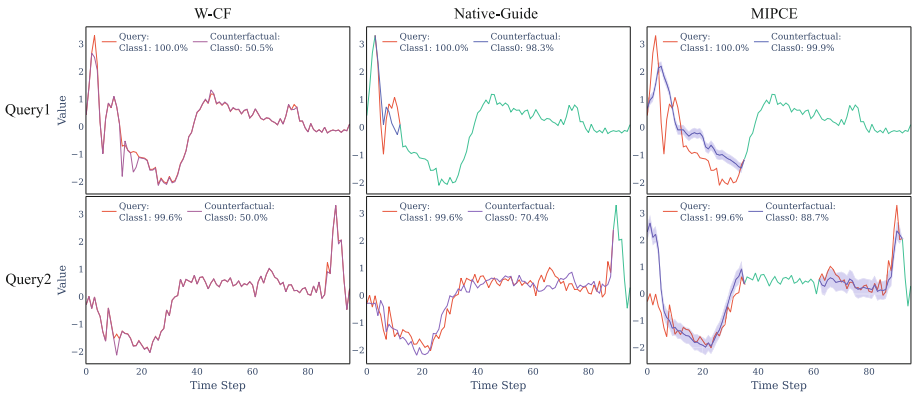


Fig. 2. Counterfactuals of the ECG200. MIPCE shows expected values as a solid line and 95% confidence intervals as fill. The color is the same as Fig. 1 right.

Table 1. Evaluation results of counterfactuals.

		L1	L2	L_Inf	OCSVM	IForest	IM1	IM2	R%-Sub
ECG200	W-CF	**0.08**	**0.25**	**0.62**	0.265	0.266	1.781	0.969	0.136
	Native-Guide	0.2	0.48	0.86	0.22	0.261	1.215	0.533	0.2
	MIPCE	0.66	0.98	1.3	**0.15**	**0.225**	**0.626**	**0.296**	**0.556**
Strawberry	W-CF	**0.11**	**0.4**	1.43	0.017	0.116	1.286	0.018	0.082
	Native-Guide	0.54	0.69	**0.81**	0.005	0.05	1.54	0.008	0.111
	MIPCE	0.83	1.06	1.44	**0.003**	**0.04**	**1.095**	**0.007**	**0.287**
GunPoint	W-CF	**0.06**	**0.18**	**0.56**	0.174	0.228	1.263	**0.035**	0.023
	Native-Guide	0.27	0.56	0.85	0.113	0.155	1.029	0.061	0.321
	MIPCE	1.4	1.65	1.79	**0.04**	**0.129**	**0.622**	0.038	**0.901**
Proximal	W-CF	**0.07**	**0.26**	**0.65**	0.014	0.064	1.185	0.013	0.204
	Native-Guide	0.31	0.54	0.83	0.003	0.057	1.208	0.011	0.247
	MIPCE	1.27	1.32	1.28	**0.0**	**0.021**	**1.016**	**0.008**	**0.692**
Wafer	W-CF	**0.01**	**0.03**	**0.08**	**0.13**	0.441	2.809	1.048	0.014
	Native-Guide	0.36	0.55	0.85	0.25	0.63	1.636	1.169	0.02
	MIPCE	0.75	0.85	1.01	0.37	**0.62**	**1.199**	**0.769**	**0.242**

Results. Figure 2 shows the counterfactuals generated by each method, along with corresponding queries, which belong to the same class. We observe that in the case of query2, MIPCE generated sparse explanations. In addition, if we examine query1 and query2 together, we can clearly interpret the important subsequences.

From a quantitative perspective, W-CF obtained the best results in terms of proximity (see Table 1). However, as seen in Fig. 2, good proximity does not necessarily correlate with high human interpretability. In addition, W-CF exhibited poor results in terms of plausibility, as it generated counterfactuals that do not

exist in the real world. Conversely, our method obtained better plausibility and substitutability scores. This suggests that MIPCE captures subsequences that are critical for classification, and generates counterfactuals that follow the data distribution.

4.2 Experiment 2: Change to the Counterfactual

In the process of continuous change, we evaluated the proximity and plausibility of instances that change the query to the counterfactual $r\%$ ($r \in \{0, 25, 50, 75, 100\}$). We used the same metrics as in Experiment 1. From a proximity perspective, it is desirable for the distance between the query and instance to increase with the changing rate of the counterfactual. From a plausibility perspective, it is desirable for instances with a change rate of approximately 50% to be OOD. These evaluations were inspired by [13].

Results. The distance from the query was observed to increase with the rate of change to the counterfactual (see Fig. 3 and Fig. 4a). Therefore, it can be said that the process of continuous change is an ideal one. In terms of plausibility, OCSVM and IForest exhibited smaller changes in their evaluation values compared to IM1 and IM2. This indicates that distance-based metrics cannot detect intermediate counterfactual instances that would not follow the data distribution. Conversely, the autoencoder's metrics judge instances close to the 50% ratio to be OOD.

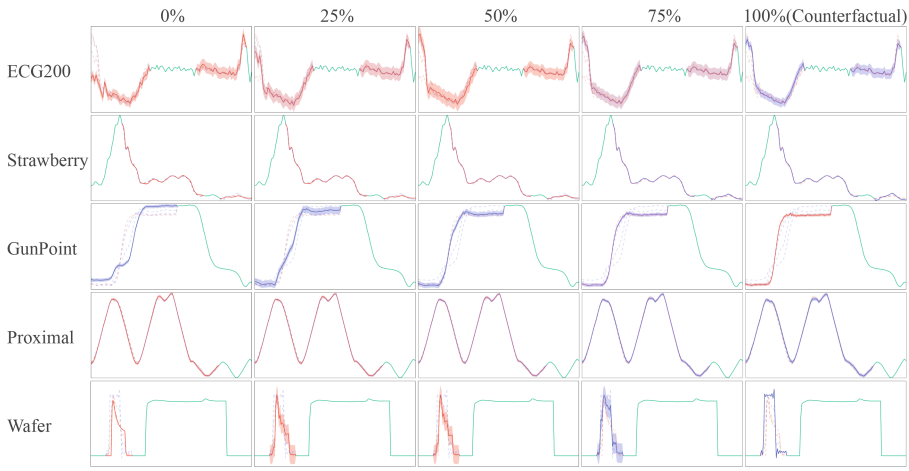
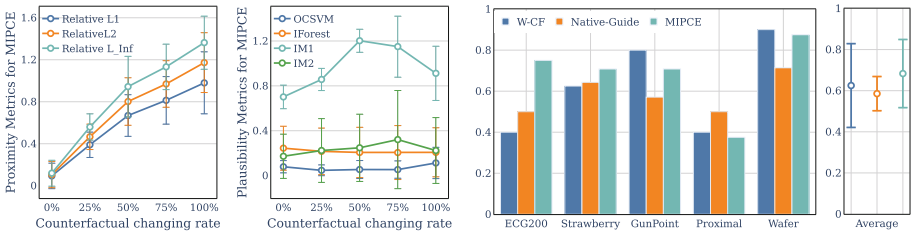


Fig. 3. Counterfactual changing of each dataset. The solid line represents the corresponding percentage, and the dashed line shows instances for other percentages. The color is the same as in Fig. 1 right.



(a) (Left)Proximity.(Right)Plausibility. **(b)** (Left)Each Datasets.(Right)Average.

Fig. 4. (a) Counterfactual change evaluation of the mean and standard deviation of the five datasets. (b) Results of user test.

4.3 Experiment 3: User Test

Present explanations generated by specific methods from W-CF, Native-Guide, and MIPCE to assess the user's understanding of the DNN decision process. Effectiveness was evaluated by measuring the ability of users to correctly predict the DNN's classification result of an unknown query. Our participants, all college students with prior knowledge of machine learning, were divided among 3 groups of approximately 6 students each. Each group was presented with 8 examples of explanations, and subsequently tested with 4 unknown queries. The results determined which explanation method is the most conducive for the user's understanding of the DNN. This experiment was inspired by [1].

Results and Discussion. As can be seen from the average accuracy (see Fig. 4b), MIPCE demonstrated superior performance on many datasets, indicating its effectiveness in enhancing user's understanding of the model. However, W-CF outperformed MIPCE on the GunPoint and Wafer datasets. Both datasets are easily recognizable to humans, and it is likely that users inferred the classification criteria from multiple queries. This suggests that for easily recognizable time series data, the informative explanations provided by MIPCE may hinder user understanding. MIPCE results were also worse on the Proximal dataset, as the generated counterfactuals altered most of the query (see Fig. 3), making it difficult for users to understand the important sequences. It is expected that this can be resolved by showing the patch division process along with the counterfactuals, or by revising the cluster merging algorithm.

5 Conclusion

For counterfactual explanations in time series classification, we propose MIPCE, which takes subsequences from an FCN and presents the counterfactual changes of the patches that contribute to classification. Quantitative evaluation results indicate that MIPCE generates more plausible counterfactuals consistent with

the data distribution compared to conventional methods. In addition, our approach is able to retrieve features that contribute to classification, indicating the potential of using them for data augmentation. Furthermore, user testing has shown the effectiveness of our method.

In the future, we will improve our method to present more effective explanations based on user feedback. One idea for improvement is to show the patch division, as well as the contribution of each patch to classification, along with the current explanation. Another direction is data augmentation. In the continuous changes of MIPCE, it is possible to obtain the classification probability and confidence level of the generated instance, which serve as indicators of how well the instance follows the data distribution. This could be used for data augmentation, and we will explore the possibility of applying our method therein.

Acknowledgements. This paper is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

1. Akula, A., Wang, S., Zhu, S.C.: Cocox: generating conceptual and counterfactual explanations via fault-lines. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 2594–2601 (2020)
2. Blei, D.M., Jordan, M.I.: Variational inference for Dirichlet process mixtures. Bayesian Anal. **1**(1), 121–143 (2006)
3. Bolei, Z., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Object detectors emerge in deep scene cnns. In: International Conference on Learning Representations (2015)
4. Byrne, R.M.: Counterfactuals in explainable artificial intelligence (xai): evidence from human reasoning. In: IJCAI, pp. 6276–6282 (2019)
5. Casella, G., Robert, C.P., Wells, M.T.: Generalized accept-reject sampling schemes. In: Lecture Notes-Monograph Series, pp. 342–347 (2004)
6. Dau, H.A., et al.: The ucr time series archive. IEEE/CAA J. Automatica Sinica **6**(6), 1293–1305 (2019)
7. Delaney, E., Greene, D., Keane, M.T.: Instance-based counterfactual explanations for time series classification. In: Sánchez-Ruiz, A.A., Floyd, M.W. (eds.) ICCBR 2021. LNCS (LNAI), vol. 12877, pp. 32–47. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86957-1_3
8. Dhurandhar, A., et al.: Explanations based on the missing: towards contrastive explanations with pertinent negatives. Adv. Neural Inf. Process. Syst. **31**, 1–12 (2018)
9. Guidotti, R., Monreale, A., Spinnato, F., Pedreschi, D., Giannotti, F.: Explaining any time series classifier. In: 2020 IEEE Second International Conference on Cognitive Machine Intelligence (CogMI), pp. 167–176 (2020)
10. Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., Muller, P.A.: Deep learning for time series classification: a review. Data Min. Knowl. Disc. **33**(4), 917–963 (2019)
11. Joshi, S., Koyejo, O., Vijitbenjaronk, W.D., Kim, B., Ghosh, J.: Towards realistic individual recourse and actionable explanations in black-box decision making systems. ArXiv arXiv:1907.09615v1 (2019)

12. Karlsson, I., Rebane, J., Papapetrou, P., Gionis, A.: Explainable time series tweaking via irreversible and reversible temporal transformations. In: 2018 IEEE International Conference on Data Mining (ICDM), pp. 207–216. IEEE (2018)
13. Kenny, E.M., Keane, M.T.: On generating plausible counterfactual and semi-factual explanations for deep learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 11575–11585 (2021)
14. Lawrence, N.: Gaussian process latent variable models for visualisation of high dimensional data. Adv. Neural Inf. Process. Syst. **16**, 1–8 (2003)
15. Liu, D.C., Nocedal, J.: On the limited memory bfgs method for large scale optimization. Math. Program. **45**(1), 503–528 (1989)
16. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: 2008 Eighth IEEE International Conference on Data Mining, pp. 413–422. IEEE (2008)
17. Liu, S., Kailkhura, B., Loveland, D., Han, Y.: Generative counterfactual introspection for explainable deep learning. In: 2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP), pp. 1–5. IEEE (2019)
18. Van Looveren, A., Klaise, J.: Interpretable counterfactual explanations guided by prototypes. In: Oliver, N., Pérez-Cruz, F., Kramer, S., Read, J., Lozano, J.A. (eds.) ECML PKDD 2021. LNCS (LNAI), vol. 12976, pp. 650–665. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86520-7_40
19. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. Adv. Neural Inf. Process. Syst. **30**, 1–10 (2017)
20. Mark T Keane, Eoin M Kenny, E.D., Smyth, B.: If only we had better counterfactual explanations: five key deficits to rectify in the evaluation of counterfactual xai techniques. In: Proceeding of the 30th International Joint Conference on Artificial Intelligence, IJCAI, pp. 4466–4474 (2021)
21. Ribeiro, M.T., Singh, S., Guestrin, C.: "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, pp. 1135–1144 (2016)
22. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (2001)
23. Singla, S., Pollack, B., Chen, J., Batmanghelich, K.: Explanation by progressive exaggeration. In: International Conference on Learning Representations (2020)
24. Titsias, M., Lawrence, N.D.: Bayesian gaussian process latent variable model. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pp. 844–851. JMLR Workshop and Conference Proceedings (2010)
25. Wachter, S., Mittelstadt, B., Russell, C.: Counterfactual explanations without opening the black box: automated decisions and the gdpr. Harv. JL Tech. **31**, 841 (2017)
26. Wang, Z., Yan, W., Oates, T.: Time series classification from scratch with deep neural networks: a strong baseline. In: 2017 International Joint Conference on Neural Networks (IJCNN), pp. 1578–1585. IEEE (2017)
27. Ye, L., Keogh, E.: Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. Data Min. Knowl. Disc. **22**(1), 149–182 (2011)
28. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2921–2929 (2016)