Postprint: Algorithms for Piecewise Feature Extraction from Functional Data

Authors: Haibo Jin, Ma Haiqiang

Date: 2019-05-10T00:00:00+00:00

Abstract

To address the issues of limited expressive capability of global statistical features and the vulnerability of salient point features to noise interference in functional data classification algorithms, this paper proposes a piecewise extraction algorithm for function curve features based on statistical depth methods. First, data smoothing techniques are utilized to preprocess discretely observed data, while incorporating first- and second-order derivative functions of the functional data. Subsequently, Mahalanobis integral depth values are computed piecewise for both the function itself and its low-order derivatives, upon which function curve feature vectors are constructed. Finally, three search schemes for selecting tuning parameters are proposed, and classification studies are performed. Experimental results on the UCR dataset indicate that, compared with existing curve feature extraction algorithms, the proposed algorithm can effectively extract function curve features and enhance classification accuracy.

Full Text

Segmental Feature Extraction for Functional Data

Jin Haibo[1], Ma Haiqiang[2] [1](Dept. of Mathematics, Taiyuan University of Science & Technology, Taiyuan 030024, China; [2]School of Statistics, Jiangxi University of Finance & Economics, Nanchang 330013, China)

Abstract: To address the limited expressive power of global statistical features and the susceptibility of salient point features to noise interference in functional data classification algorithms, this paper proposes a segmental feature extraction algorithm for functional curves based on statistical depth methods. First, data smoothing techniques are applied to preprocess discrete observations while introducing first and second order derivative functions of the functional data. Then, Mahalanobis integral depth values are computed segmentwise for the function itself and its lower-order derivatives, based on which feature vectors for functional curves are constructed. Finally, three search schemes for selecting tuning parameters are presented, and classification studies are conducted. Experiments on UCR datasets demonstrate that compared with other existing curve feature extraction algorithms, the proposed algorithm can effectively extract functional curve features and improve classification accuracy.

Key words: functional data; segmental feature; depth function; functional data classification

0 Introduction

Over the past three decades, rapid advances in science and technology have dramatically enhanced our ability to acquire and store data. In many real-world domains, there is increasing need to handle data with functional characteristics such as temporal and spatial properties, including financial data in economic activities, sensor data from industrial equipment, and meteorological data in environmental science. These data are often noisy discrete observations that require effective re-expression for analysis, after which appropriate data mining techniques such as classification or clustering can be applied. By treating sequential data as functional data [1], we can fully leverage the excellent properties and characteristics of functions to significantly enhance the depth and precision of data mining.

Functional data are essentially infinite-dimensional, with sample units being random curves in infinite-dimensional function space—realizations of stochastic processes. Functional data analysis methods primarily employ techniques from functional analysis to model such data, emphasizing the functional nature of the data by treating observations over a time interval as a whole without imposing assumptions on internal dependency structures. For different observational units, functional data analysis permits the use of different sampling techniques to obtain sparse or dense observations at various time points.

The inherent infinite-dimensional nature of functional data inevitably poses significant challenges for practical analysis. Direct data mining on functional data incurs substantial computational costs [2], making dimensionality reduction or feature extraction necessary [3-5]. Once low-dimensional features are obtained, mature data mining techniques can be applied to improve computational efficiency. Notably, different feature extraction methods should be employed for functional data depending on the specific analysis purpose and application domain. For classification problems, features correlated with class labels should be extracted to enhance classification accuracy [4,5].

Numerous experts and scholars have conducted extensive research on functional data classification. Considering the close relationship between feature extraction and classification algorithms—particularly since the quality of feature extraction must be evaluated through classification results—we review previous work by discussing both classification algorithms and their employed feature extraction techniques. Alonso et al. [6] reconstructed discriminant variables based on distances between functional curves and class mean curves, then applied multivariate classification techniques such as Linear Discriminant Analysis (LDA) or k-Nearest Neighbors (KNN). In addition to the function itself, their algorithm utilized multiple derivative orders to construct discriminant variables, with experiments demonstrating that features constructed from first and second derivatives significantly reduce misclassification rates.

Torrecilla et al. [7] proposed an iterative algorithm for functional feature selection via recursive maxima hunting (RMH). They first computed distance correlation between random functions and class variables to identify salient points corresponding to maximum coefficient values, then recursively searched for salient points after removing their influence on subintervals. Dai et al. [8] introduced a Bayesian classification algorithm based on likelihood ratios and theoretically proved its "perfect classification" property. They first projected functional data to obtain multiple independent principal component scores, then performed nonparametric estimation of the probability density functions for these scores, and finally completed Bayesian classification using likelihood ratio formulas.

Mosler et al. [9] employed a two-step transformation method: first using segmentwise integral values of functions and their first derivatives as feature vectors, then applying multivariate depth functions to map these vectors to a two-dimensional DD-plot, and finally using nearest neighbor classification and DD-procedures for classification. Li et al. [10] first used F-statistics to identify curve salient points and their neighboring subintervals, then extracted curve features using LDA, and finally applied Support Vector Machines (SVM) for classification—particularly suitable for spatially heterogeneous or irregularly sampled curve data. Fraiman et al. [11] defined curve features using a set of functions and applied them to classification, regression, and principal component analysis. Rossi et al. [12] detailed the application of Support Vector Machines in functional data classification.

Domestic scholars Ma Chen et al. [13] proposed a Fast Feature Selection (FFS) method for functional data combining principal component analysis and minimum convex hull methods, which can rapidly obtain stable feature subsets with excellent practical performance. Su Benyue et al. [14] employed functional data analysis methods to functionalize human periodic behavioral data collected by wearable motion capture systems, accurately defining data continuity and periodicity, and finally used Support Vector Machines for dynamic behavior classification and recognition based on differences in curve characteristics within one period across different behaviors.

1 Segmental Feature Extraction (SFE) Algorithm

Considering the limited expressive power of global statistical features and the vulnerability of salient point features to noise interference, this paper proposes a segmental feature extraction algorithm based on statistical depth functions (SFE). Leveraging the smoothing characteristics of functional data, the proposed algorithm utilizes not only the properties of the function itself but also segmental features of multiple derivative orders, thereby comprehensively characterizing variation patterns in functional data. Experiments on multiple UCR datasets verify the practical effectiveness of the proposed algorithm in functional data classification applications.

1.1 Problem Definition and Algorithm Flow

For descriptive convenience, consider a binary classification problem for functional data. Let $\{X_i(t), t \in [0,T]\}_{i=1}^{n}$ be a time-continuous stochastic process from probability space $(\Omega, \mathcal{F}, P)$, where $X_i(t)$ represents a realization (or trajectory) of this process. Let $\{(X_i(t), Y_i)\}_{i=1}^{N}$ be a dataset of function-class label pairs, where $Y_i \in \{0,1\}$ is a classification variable indicating that these functions (trajectories) originate from two different populations P_0 and P_1, with $Y_i = 0$ representing class P_0 and $Y_i = 1$ representing class P_1. The classification problem involves inferring the population P_0 or P_1 to which an unknown function object $X_{test}(t)$ belongs. Essentially, feature extraction aims to find a mapping of the form $d : \mathcal{F}_R \to \mathbb{R}^m$ that transforms the infinite-dimensional functional data classification problem into a finite-dimensional classification problem in $\mathbb{R}^m$, thereby avoiding the infeasibility of classification in infinite-dimensional function space.

[Figure 1: see original paper] illustrates the main workflow of the SFE algorithm for data processing. Figure 1(a) shows noisy discrete sequence data, which after smoothing yields continuous functional data as shown in (b). Lower-order derivative functions are then computed as shown in (c), displaying the original function, first derivative, and second derivative from top to bottom. These three types of functions are then segmented as shown in (d). Finally, statistical depth values are calculated for each segment and combined to form depth value vectors as shown in (e), yielding the function' s feature vector.

1.2 Data Smoothing

If the observed samples under study are noisy data sequences satisfying the model $Y(t) = X(t)+\varepsilon(t)$, where residuals $\varepsilon(t)$ are independent, linear smoothing methods can be applied to obtain the original function $\hat{X}(t) = \sum_{j=1}^{m} s_{ij} Y(t_j)$, where s_{ij} represents the weight of point t_j relative to t_i, and $S = (s_{ij})$ can be viewed as a smoothing matrix.

Currently, two main linear smoothing methods exist for recovering the original function. One approach uses a linear combination of basis functions $\{\phi_k\}_{k=1}^{K}$ to approximate $X(t)$, i.e., $X(t) \approx \sum_{k=1}^{K} c_k \phi_k(t)$. Here, a sufficiently large number K of basis functions is selected, i.e., $K \approx n$. In the proposed algorithm, a set of B-spline basis functions is used to approximate the original function.

Another method employs nonparametric kernel smoothing techniques [2], using the Nadaraya-Watson estimator with smoothing matrix S where s_{ij} =

$K\left(\frac{t_i-t_j}{h}\right) / \sum_{k=1}^{n} K\left(\frac{t_k-t_j}{h}\right)$. Here $K(\cdot)$ is a kernel function, typically a Gaussian kernel, and the optimal bandwidth parameter h can be obtained through cross-validation.

To compare the differences between these smoothing methods, [Figure 2: see original paper] provides illustrative diagrams of B-spline curve smoothing and nonparametric kernel smoothing. The curve data originates from the 34th curve segment in the GunPoint dataset, containing 21 data points. The number of B-spline basis functions and the kernel bandwidth parameter are computed through cross-validation from a set of candidate values. In this example, [Figure 2: see original paper] shows that the first smoothing method⋯

1.3 Statistical Depth Function

Statistical depth functions and related quantile functions enable nonparametric description and structural analysis of multivariate data. Various types of depth functions exist based on different centrality concepts and definitions. For functional data, depth functions can also be defined to characterize the centrality of a curve (process) relative to a sample of curves [15,16].

Consider curves $X_i(t), i = 1, \ldots, n$ from sample $\{X_i(t)\}_{i=1}^{n}$. This paper employs the Mahalanobis integral depth function defined as follows [16]:

$$FMD(X_i, P) = \int_0^T \left(1 - \left|\frac{1}{2} - F_{X(t)}(X_i(t))\right|\right) dt$$

where

$$F_{X(t)}(x) = \frac{1}{n} \sum_{k=1}^{n} I\{X_k(t) \leq x\}$$

and $I\{\cdot\}$ denotes the indicator function.

Equation (4) essentially represents a generalization of depth functions to functional data. Similar to order statistics for one-dimensional random variables, the Mahalanobis integral depth function primarily characterizes the positional information of a curve within the entire dataset. The Mahalanobis integral depth function is not only computationally simple but also possesses excellent robustness properties. Additionally, based on Equation (4), numerous statistics for functional data can be constructed, such as ranks and trimmed means for functional data, thereby overcoming the influence of outliers and yielding more accurate and reliable analysis results.

1.4 Segmental Feature Extraction

Assume $X_i^{(1)}$ and $X_i^{(2)}$ are function samples from populations P_0 and P_1, respectively. For any $X_i \in \{X_i^{(1)}, X_i^{(2)}\}$, let $D_0^i X$, $D_1^i X$, and $D_2^i X$ represent the function itself, its first derivative, and second derivative, respectively. These three types of functions describe the position, slope variation, and concavity/convexity properties of functional curves. If we consider the transformation $f : \mathcal{F}_R \to \mathbb{R}^6$ defined as:

$$f(X_i) = \left(f(D_0^i X^{(1)}, D_0^i X^{(2)}), f(D_1^i X^{(1)}, D_1^i X^{(2)}), f(D_2^i X^{(1)}, D_2^i X^{(2)})\right)$$

where $f(D_p^k X^{(1)}, D_p^k X^{(2)})$ represents the statistical depth value of $D_p^i X$ relative to $D_p^k X^{(1)}$ and $D_p^k X^{(2)}$, and the integral depth function defined by Equations (4) and (5) is applied here.

Noting that the above transformation targets global statistical features correlated with class labels, to enhance local feature expressive power, we consider segmentwise application of the f transformation. For descriptive convenience, we analyze $D_0^i X$; $D_1^i X$ and $D_2^i X$ can be analyzed similarly.

Divide the domain $[0, T]$ into N_l equally spaced subintervals: $[0, T/N_l), [T/N_l, 2T/N_l), \ldots, [(N_l - 1)T/N_l, T]$. Represent the function segment on each subinterval using X_{ij} for $j = 1, \ldots, N_l$, then apply the f transformation to each segment. Similarly, divide $D_1^i X$ and $D_2^i X$ into N_s and N_c segments, respectively. In summary, we obtain the mapping from function space to feature space:

$$\Phi : \mathcal{F}_R \to \mathbb{R}^{N_l + N_s + N_c}$$

where $N_l + N_s + N_c \geq 2$.

1.5 Selection of Tuning Parameters

The three most important parameters in the SFE algorithm are the segment numbers N_l, N_s, and N_c. The selection of these parameter values directly affects subsequent classification algorithm performance, yet theoretical guidance for their optimal selection remains lacking [9]. Experiments on different datasets indicate that satisfactory results can typically be achieved with segment numbers not exceeding 10, but exhaustive search for the optimal combination of three parameters remains time-consuming. Therefore, heuristic search strategies are needed to improve efficiency.

Two simplified schemes are considered here. The first is independent selection, which searches for three parameters separately to determine their optimal segment numbers n_l, n_s, and n_c. This method is simple to implement and the search tasks can be parallelized. The second scheme is stepwise selection: first assume $(N_s, N_c) = (0, 0)$ and search for the optimal segment number n_l for N_l; then fix $N_l = n_l$ and search for the optimal segment number n_s for N_s under this condition; finally fix $(N_l, N_s) = (n_l, n_s)$ and search for the optimal segment number n_c for N_c, ultimately obtaining the optimal parameter combination (n_l, n_s, n_c). Evidently, both independent selection and stepwise selection methods are far more efficient than exhaustive search.

1.6 Algorithm Pseudocode

Based on the above discussion, the pseudocode for the segmental feature extraction algorithm SFE based on depth information is presented as Algorithm 1. For simplicity, only the core code is listed.

Algorithm 1 requires explanation: X and Y represent the function sample set and corresponding class labels, respectively; N_l, N_s, and N_c represent the segment numbers for the function and its first and second derivatives. Lines 1-7 compute statistical depth values for function segments, where line 3 divides the sample set into two class subsets, and lines 5-6 compute depth values according to Equation (4). Lines 8-14 perform function segmentation and compute segmental feature values, where line 10 segments the function, and lines 11-13 cyclically call the depth function to compute statistical depth values for each segment. Lines 15-23 constitute the main algorithm procedure: line 16 smooths the original observed data using Equation (2), lines 17-18 obtain the first and second derivatives of the function, lines 19-21 compute segmental feature values for the function and its derivatives, and line 22 combines the feature values of the function and its derivatives into a feature matrix.

Algorithm 1: Segmental Feature Extraction Algorithm Input: X, Y, N_l, N_s, N_c
Output: features (feature matrix)

1. function depth(X_i, Y)
2. `$d \leftarrow$ matrix(size($X_i$), 2)`
3. `$\{X_i^{(1)}, X_i^{(2)}\} \leftarrow$ groupBy($X_i$, $Y$)`
4. `for($k$ in 1:size($X_i$)) {`
5. `    $d(k,1) \leftarrow FMD(X_i[k], X_i^{(1)})$`
6. `    $d(k,2) \leftarrow FMD(X_i[k], X_i^{(2)})$ }`
7. `return $d$`
8. function seg_features(X, Y, n)
9. `$fx \leftarrow$ matrix(size($X$), $2n$)`
10. `$\{X_1, \ldots, X_n\} \leftarrow$ segment($X$, $n$)`
11. `for($i$ in 1:$n$) {`
12. `    $fx[,(2i-1)] \leftarrow$ depth($X_i$, $Y$)[,1]`
13. `    $fx[,2i] \leftarrow$ depth($X_i$, $Y$)[,2] }`
14. `return $fx$`
15. procedure SFE(X, Y, N_l, N_s, N_c)
16. `$D_0X \leftarrow$ smooth($X$)`

17. `$D_1X \leftarrow$ deriv($D_0X$, 1)`
18. `$D_2X \leftarrow$ deriv($D_0X$, 2)`
19. `$f_{D_0X} \leftarrow$ seg_features($D_0X$, $Y$, $N_l$)`
20. `$f_{D_1X} \leftarrow$ seg_features($D_1X$, $Y$, $N_s$)`
21. `$f_{D_2X} \leftarrow$ seg_features($D_2X$, $Y$, $N_c$)`
22. `features $\leftarrow [f_{D_0X}\ f_{D_1X}\ f_{D_2X}]$`
23. `return features`

2 Experiments

2.1 Dataset Description

To compare classification performance across different algorithms, this paper selects six datasets from the UCR standard time series dataset repository [17] for experiments. These datasets exhibit complex curve characteristics, posing significant challenges for classification. Each dataset contains independent training and test sets, with detailed descriptions provided in . [Figure 3: see original paper] displays sample curves from two classes in the WormsTwoClass dataset.

TABLE:1 Experimental datasets (table content preserved as in original)

2.2 Experimental Design

This paper proposes a segmental feature extraction algorithm SFE for functional data, which obtains low-dimensional features before applying mature classification algorithms. To verify the generality of the proposed algorithm, experiments employ multiple combinations of SFE with classification algorithms. LDA denotes Linear Discriminant Analysis, SVM denotes Support Vector Machine with radial basis kernel function (parameters $c = 8$, $\gamma = 0.5$), and RF denotes Random Forest classification method with default classifier parameters. For fair comparison, classification algorithm parameters remain unchanged across different datasets.

Algorithm code is implemented in R language, with primary third-party packages including the functional data analysis package `fda.usc` and the classification and regression training package `caret`, the latter used for building classification models including model training and classification prediction. The training set is used for model fitting and parameter optimization via K-fold cross-validation, where K is set to 5 or 10 depending on training set size. After model establishment, independent test sets evaluate model performance, with classification accuracy serving as the evaluation metric.

2.3 Results and Analysis

TABLE:2 lists classification accuracies across datasets under different classification methods. The first two columns present results from two simple nonparametric classification models [17], which serve as baseline data for evaluating the proposed algorithm' s performance. 1NN denotes nearest neighbor classification using Euclidean distance as similarity measure, while 1NN-DTW denotes nearest neighbor classification using Dynamic Time Warping distance. For non-time-aligned curve samples, 1NN-DTW classification yields better results than simple 1NN, as demonstrated on Earthquakes and WormsTwoClass datasets. The remaining eight columns show classification accuracies under four classification algorithms with and without curve segmentation, where non-segmentation corresponds to SFE algorithm parameters N_l, N_s, and N_c set to 1. Classification accuracy under segmentation conditions heavily depends on segment numbers, with the table presenting optimal results under best segment numbers. Notably, due to different parameter search strategies, the obtained segment numbers are not unique and may cause classification result variations.

TABLE:2 Classification accuracy comparison of different classification methods (table content preserved as in original)

Analysis of results in Table 2 yields the following conclusions:

a) Without curve segmentation, classification results obtained using the SFE algorithm differ little from baseline results in most cases, demonstrating that statistical depth values are highly effective as curve features. For Ham and Earthquakes datasets, classification accuracies of the three classification algorithms improve by an average of 16.1% and 12.1% compared to 1NN baseline results.

b) With curve segmentation, higher-dimensional features are obtained from curves, and classification accuracies under all classification algorithms exceed those without segmentation. This is particularly evident on GunPoint, BeetleFly, and Herring datasets, where classification accuracies under the three classification algorithms improve by an average of 5.33%, 13.3%, and 10.9% compared to non-segmentation cases. Additionally, the best classification accuracies on GunPoint and WormsTwoClass datasets are 95.3% and 72.9%, respectively—improvements of 23.3% and 14.9% over the best results reported in literature [8] for the same datasets.

c) For fair comparison of multiple classification algorithms across different datasets without highlighting SFE algorithm effects, no classification algorithm parameters were optimized and no excessive feature preprocessing was performed, which may lead to unexpected classification results. For instance, SVM classification results on the Ham dataset suffer from overfitting, yielding lower test set accuracy than expected. In practical data analysis, these issues can be avoided through feature preprocessing techniques and classification algorithm parameter optimization.

2.4 Comparison with Other Feature Extraction Methods

Common dimensionality reduction methods for functional data include Principal Component Analysis (PCA) [1] and Partial Least Squares (PLS) [4,5]. Authors in [6] proposed the DFM (Distance to Functional Mean) method, whose main idea extracts features from functions and their derivatives according to:

$$d_p = \int_0^T \left(|X(t) - \bar{X}^{(1)}(t)|^p - |X(t) - \bar{X}^{(2)}(t)|^p\right) dt$$

where d_p represents extracted real-valued features, $\bar{X}^{(1)}(t)$ and $\bar{X}^{(2)}(t)$ denote mean function curves for positive and negative classes, respectively, and p is typically 1 or 2.

To compare performance of multiple feature extraction methods, this paper conducts experiments on the six aforementioned datasets, with LDA used as the subsequent classification algorithm for all methods. Algorithm code is implemented in R language. Notably, PCA determines principal component numbers based on cumulative variance contribution rate, PLS obtains component numbers through cross-validation, and DFM extracts features from original functions, first derivatives, and second derivatives according to Equation (8). All methods perform functional representation and smoothing on original data. Classification accuracies on test datasets are shown in **TABLE:3**.

TABLE:3 Classification accuracy comparison of different feature extraction methods (table content preserved as in original)

Comparing LDA classification results in Table 3 with those in Table 2 reveals that among 18 total analysis results, only three cases achieve higher classification accuracy than the proposed method: DFM on the Ham dataset, PCA on the Herring dataset, and PLS on the Earthquakes dataset. This demonstrates that the proposed SFE method is generally superior to the other three feature extraction methods. Additionally, since PLS considers correlations between sample data and class variables, it extracts higher-quality features and achieves better classification results than PCA, though PLS requires longer running time.

3 Conclusion

Classification is a crucial research direction in functional data analysis, where effective extraction of low-dimensional features from functional data is key. The proposed algorithm segments functions and derivative curves, transforms infinite-dimensional functions into low-dimensional feature vectors based on statistical depth methods, and then applies standard classification algorithms. This approach avoids deficiencies in global feature and salient point feature representation. Experimental results on multiple datasets validate the effectiveness of the SFE algorithm. Further consideration of three issues: (a) Handling non-time-aligned sample curves, such as through landmark registration, would significantly improve subsequent functional representation and analysis; (b) Current algorithm uses equally spaced segmentation intervals—whether heuristic strategies can be proposed to adaptively determine non-equidistant subintervals for extracting more discriminative class features; (c) More transformation forms can be considered for functional feature mapping, where the statistical depth values (centrality measures) used in this paper can be extended to multiple definitions. These three points represent the focus of future work.

References

[1] Ramsay J O, Silverman B W. Functional data analysis [M]. 2nd ed. New York: Springer, 2005.

[2] Ferraty F, Vieu P. Nonparametric functional data analysis: theory and practice [M]. New York: Springer, 2006.

[3] Meng Yinfeng, Liang Jiye. Robust principal analysis in the classification for functional data [J]. Journal of Chinese Computer Systems, 2016, 37(7): 1499-1503.

[4] Preda C, Saporta G, Lévéder C. PLS classification of functional data [J]. Computational Statistics, 2007, 22(2): 223-235.

[5] Delaigle A, Hall P. Methodology and theory for partial least squares applied to functional data [J]. Annals of Statistics, 2012, 40(2012): 322-352.

[6] Alonso A M, Casado D, Romo J. Supervised classification for functional data: a weighted distance approach [J]. Computational Statistics and Data Analysis, 2012, 56(7): 2334-2346.

[7] Torrecilla, José L, Suárez et al. Feature selection in functional data classification with recursive maxima hunting [C]// Advances in Neural Information Processing Systems. 2016: 4835-4843.

[8] Dai X, Myuller H G, Yao F. Optimal bayes classifiers for functional data and density ratios [J]. Biometrika, 2017, 104(3): 545-560.

[9] Mozharovskyi P, Mosler K. Fast DD-classification of functional data [J]. Statistical Papers, 2017, 58(4): 1055-1089.

[10] Li B, Yu Q. Classification of functional data: a segmentation approach [J]. Computational Statistics and Data Analysis, 2008, 52(10): 4790-4799.

[11] Fraiman R, Gimenez Y, Svarc M. Feature selection for functional data [J]. Journal of Multivariate Analysis, 2016, 146: 191-208.

[12] Rossi F, Villa N. Support vector machine for functional data classification [J]. Neurocomputing, 2006, 69(7-9): 730-742.

[13] Ma Chen, Wang Wenjian, Jiang Gaoxia. Fast feature selection for functional data [J]. Pattern Recognition and Artificial Intelligence, 2017, 30(9): 822-832.

[14] Su Benyue, Jiang Jing, Tang Qingfeng et al. Human dynamic action recognition based on functional data analysis [J]. Acta Automatica Sinica, 2017, 43(5): 866-876.

[15] Sguera C, Galeano P, Lillo R. Spatial depth-based classification for functional data [J]. Test, 2014, 23(4): 725-750.

[16] Serfling R, Wijesuriya U. Depth-based nonparametric description of

functional data, with emphasis on use of spatial depth [J]. Computational Statistics and Data Analysis, 2017, 105: 24-45.
[17] Chen Y, Keogh E, Bing H et al. The UCR time series classification archive [DB/OL]. (2015) [2018-11-01]. http://www.cs.ucr.edu/~eamonn/time_series_data.

Note: Figure translations are in progress. See original paper for figures.