



Dynamic Sub-Sequence Warping: A Representation-Based Similarity Measure for Long Time Series

Zhou Zhou¹ · Gang Huang² · Laura Dawkins³

Received: 23 April 2025 / Revised: 3 November 2025 / Accepted: 25 November 2025 / Published online: 2 February 2026
© The Author(s) 2026

Abstract

Dynamic time warping (DTW), a typical elastic similarity measure that compares one-to-many points, has been proven effective for various time-series data mining tasks. However, it requires a quadratic time complexity $O(n^2)$ proportional to the length of time-series data, which undermines its applications involving long time series. In this paper, a representation-based similarity measure called Dynamic Sub-Sequence Warping (DSSW) is proposed. Instead of working on the raw data directly, we perform data representation to extract the distributional features of time series. Then, the similarity between two time series is measured by aligning the corresponding sub-sequences composed of the extracted features. We evaluate the proposed method through a supervised learning task on extensive real-world datasets. The results show that DSSW outperforms the prevalent DTW-based methods in terms of precision, and achieves one order of magnitude faster execution time on average compared with DTW.

Keywords Time series · Similarity measure · Representation · Dynamic time warping

1 Introduction

Time series is one of the most common forms of data generated from almost all scientific and industrial domains. Consequently, the last two decades have seen increasing interest in knowledge discovery from time series data, namely, extracting implicit patterns and knowledge from time series to assist in optimal decision-making. This has also triggered the development of various data mining techniques for the processing, analysis, and storage of time series, such as query-by-content [35], anomaly detection [29], clustering [43], and classification [46]. Directly or indirectly, a notion

of similarity is a necessity during the implementation of these techniques.

Quantifying the similarity between time series is, however, a challenging task due to the temporally ordered nature of time series. This motivates the continuous exploration of accurate and robust similarity measures for time series data mining. Many similarity measures have been proposed in the literature, among which Dynamic Time Warping (DTW) [5] is the most prominent method. It has been proven that DTW could, on average, provide hard-to-beat results in different time series data mining tasks [28]. However, the quadratic time complexity of DTW due to the dynamic programming process undermines its applications in many cases involving long time series [17].

As a result, many works have been conducted to simplify the construction of the dynamic alignment in DTW. Typical strategies include restricting the warping path [19] and adding lower bounds [21]. Nevertheless, the retrieval of the optimal path may not be guaranteed via integrating these extra constraints, thus affecting the accuracy of the similarity measure [18]. The acceleration of DTW via data representation has recently started attracting attention as it can largely reduce the computation cost by computing DTW in a low-dimensional space. A proven side-effect of such smoothing operations is to enhance the accuracy of

✉ Gang Huang
huanggang@zju.edu.cn

Zhou Zhou
z.zhou2@exeter.ac.uk

Laura Dawkins
laura.dawkins@metoffice.gov.uk

¹ Department of Engineering, University of Exeter, Exeter EX4 4QF, UK

² College of Electrical Engineering, Zhejiang University, Hangzhou 310027, People's Republic of China

³ Met Office, Fitzroy Road, Exeter EX1 3PB, UK

downstream data mining tasks, as they reduce the effects of noisy data [31]. This further motivated the exploration of representation-based DTW methods. Some representation methods have been explored, such as Control Chart Approximation [40], Local Extrema Approximation [36] and Principal Component Analysis [30]. Despite a few recent works in this direction, they still fail to provide promising results in the downstream time-series data mining tasks due to the insufficient extraction of representative features from the raw time series.

In this paper, a new elastic distance measure named dynamic sub-sequence warping (DSSW) is proposed to compute DTW distance in a low-dimensional feature space. The dimension reduction of the raw time series is achieved by three-dimensional piecewise cloud representation (TDPCR) [42], with the key motivation of retaining sufficient distributional features of the raw time series to enhance the alignment of time series in the feature space. Specifically, the raw time series is first transformed into a series of sub-sequences comprising three-dimensional cloud features, which keep the fundamental shape patterns of the raw time series. Then, the similarity between the approximated time series is measured by aligning the corresponding sub-sequences comprised of the extracted features. Thus, the problem of feature shift between sub-sequences can also be overcome. The main objective of this paper is to enhance the efficiency of the raw DTW. Due to the proven positive side-effect of smoothing, we also expect that the proposed method can improve accuracy in downstream tasks. To systematically evaluate the proposed method, we test it on typical supervised learning tasks and compare its computational efficiency and accuracy with state-of-the-art DTW-based solutions. The experimental results on a wide range of real-world datasets [11] demonstrate the effectiveness of the proposed DSSW method.

The rest of this study is organised as follows. Section 2 reviews the works on time-series similarity measures and dynamic time warping related to this paper. Section 3 briefly introduces the principle of the DTW algorithm and the concepts of the three-dimensional piecewise cloud representation. Section 4 provides a detailed description of the proposed DSSW method. Details of the experimental results and analysis are presented in Sect. 5. Finally, the conclusions are drawn in Sect. 6.

2 Related Works

2.1 Time-Series Similarity Measure

Many similarity measures have been proposed in the literature, which can be roughly classified into lock-step

measures and elastic measures [1]. L_p -norms (e.g., the Manhattan distance, the Euclidean distance, and the Maximum distance with respect to $p = 1$, $p = 2$, and $p = \infty$, respectively) are the most widely used lock-step measures because of their simplicity and computational efficiency. However, L_p -norms and other lock-step measures are sensitive to the shape variations in the time axis, such as shifting and scaling. Besides, time series are restricted to be of equal length as a prerequisite. Consequently, most time series research has concentrated on alternative elastic measures to mitigate the shortcomings of lock-step measures. Typical elastic measures include Dynamic Time Warping (DTW) [5], Longest Common Sub-Sequence (LCSS) [12], and Edit Distance with Real Penalty (ERP) [10]. Among them, DTW is the most prominent method that aligns two time series through dynamic programming to allow the comparison of one-to-many points. Although DTW is not sensitive to the shape variations in the time axis, the cost of quadratic time complexity undermines its applications in many cases involving long time series [17]. As a result, accelerating the computation speed of DTW without compromising on accuracy has always been a hot topic.

Previous works have been conducted mainly from two perspectives to improve the efficiency of DTW: (1) integrating extra constraints and (2) reducing the dimension of raw time series. The former can be realised via warping constraints and lower bounding. While the primary objective of incorporating a warping constraint is to prevent pathological warping, it also accelerates DTW computation as a secondary effect [37]. The Sakoe-Chiba band [38] and the Itakura parallelogram [22] are the most common constraints, which apply a band to restrict the warping path in the DTW distance matrix. Many recent works have followed this thread, for instance, considering time weights [28] or integrating adaptive penalty functions [32]. However, they are built on the assumption that a decent warping path is close to the diagonal of the DTW distance matrix. Thus, the retrieval of the optimal path may not be guaranteed. Besides, DTW can still be time-consuming if it is repeatedly utilised, such as indexing similar time series from a large database [45]. To this end, lower bounding is applied to prune off unpromising candidates via a heuristic lower bound; that is, the DTW distance between two time series is computed only if their lower bounding distance is lower than a best-so-far distance [41, 44]. The primary objective of these studies is to establish a tighter lower bound than the LB_Keogh bound [24], which is by no means trivial. Instead of applying extra constraints, the other idea is to perform dimension reduction on time series via data representation and compute the DTW distance in the low-dimensional feature space, which will be discussed in the following subsection.

2.2 Time-Series Representation for Dynamic Time Warping

Latency-sensitive analytics in modern computing environments (e.g., IoT and cyber-physical systems) [34, 39] require similarity search and learning to operate under tight time and energy budgets. A practical way to meet these constraints is to reduce the dimensionality of raw time series prior to alignment or modelling, so that computation and memory scale with the number of representation elements rather than the original length. Time-series representation methods address this need by either projecting an entire sequence into a lower-dimensional space or extracting local, piecewise features that preserve discriminative structure, enabling efficient use of elastic measures such as DTW.

Various representation algorithms have been proposed to reduce the dimension of raw time series. One of the useful ways of grouping these algorithms is based on the granularity of features that the techniques attempt to find. There are two basic categories [16]: Non-data adaptive representations project the whole time series into a low-dimensional space while minimising the global reconstruction error; Data adaptive representations, on the contrary, construct a representation considering the local properties of data. A typical non-data adaptive method is the Discrete Fourier Transformation (DFT), which represents the original time series by its main frequency coefficients in the frequency domain [2]. Other global models, such as ARMA [33], have also been used to estimate the complicated underlying model given a time series. However, these global transformations usually depend on specific assumptions about the data-generating processes without the consideration of local patterns, which contain crucial discriminatory features for time series similarity evaluation. Piecewise representation is the most prevalent technique when it comes to data adaptive representations. Within this category, the original time series is divided into continuous segments from which local patterns are extracted. For example, adaptive piecewise constant approximation (APCA) [23, 47] represents a time series by continuous constant value segments of varying lengths. Piecewise linear approximation (PLA) [8, 48] splits a time series into representative segments and fits each segment with a polynomial function. Symbolic aggregate approximation (SAX) [3, 9] discretises the raw time series into symbols. Despite the various representation methods that have been proposed, a few of them could be integrated with DTW to support time series alignment in the feature space. For instance, one of the foundational works PDTW [25] transforms time series into frames and performs DTW using the mean value of the frames. PSADTW [7] represents time series by multiple statistical features and computes the DTW distance in the feature space. 3D-DTW [40]

proposes an adaptation of DTW in 3-D space based on the control chart approximation. However, the performance of these methods relies on the manual configuration of multiple hyperparameters. LE-DTW [26] develops a parameter-free distance measure based on local extrema approximation. Following this work, ESDTW [36] aligns the local shapes of both maxima and minima together to maintain the temporal relationship of the time series. It achieves state-of-the-art performance on typical time series classification tasks. However, both LE-DTW and ESDTW are less efficient in dealing with time series containing continual fluctuations as the computation cost is proportional to the number of local extrema. In contrast, the proposed method aligns time series based on their distributional features, enabling it to maintain efficiency when handling fluctuating time series.

3 Preliminaries

This section introduces the methodologies and definitions related to this work. Specifically, we will introduce the fundamentals of dynamic time warping and the time-series representation method upon which the proposed method is built.

3.1 Dynamic Time Warping

Dynamic time warping is generally used to align two sequences that are out of sync. Let $T_1 = (a_1, a_2, \dots, a_m)$ and $T_2 = (b_1, b_2, \dots, b_n)$ be two series of length m and n , G be an $m \times n$ distance matrix where element (i, j) represents the alignment (distance) between a_i and b_j , i.e., the i th and j th elements of T_1 and T_2 , respectively. A warping path $P = ((A_1, B_1), \dots, (A_s, B_s))$ is a traversal of G where (A_i, B_i) ($A_i \in [1, m], B_i \in [1, n], i \in [1, s]$) is an index pair. A valid path should satisfy the following conditions [6]:

- Boundary condition. The path should start from $(1, 1)$ and end with (m, n) , i.e., $(A_1, B_1) = (1, 1)$ and $(A_s, B_s) = (m, n)$.
- Monotonicity condition. The path can not go back in time, i.e., $A_l \leq A_{l+1}$ and $B_l \leq B_{l+1}$.
- Continuity condition. The path only contains transitions from adjacent elements in time, i.e., $A_{l+1} - A_l \leq 1$ and $B_{l+1} - B_l \leq 1$.

Suppose that $d(a_{A_i}, b_{B_i})$ refers to the element-wise distance between a_{A_i} and b_{B_i} , the DTW distance of a warping path P is defined as the accumulation of the element-wise distance, i.e.,

$$DTW_P[A_s, B_s] = \sum_{i=1}^s d(a_{A_i}, b_{B_i}) \quad (1)$$

$$DTW_{P^*}[A_i, B_i] = d(a_{A_i}, b_{B_i}) + \min\{DTW_{P^*}[A_{i-1}, B_i], DTW_{P^*}[A_i, B_{i-1}], DTW_{P^*}[A_{i-1}, B_{i-1}]\} \quad (2)$$

Here, P^* represents the optimal warping path, obtained by $P^* = \arg \min_{P \in \mathcal{P}} DTW_P$, with $\mathcal{P}$ denoting the space of all valid warping paths.

3.2 Three-Dimensional Piecewise Cloud Representation

The cloud model was first proposed in Li [27] to describe the randomness and fuzziness of concepts in natural languages. It has also been proven to be an effective technique in reducing the dimension of time series [15]. Recently, TDPCR proposed in Si et al. [42] extends the original normal cloud model by considering additional variation information of the raw time series. In this section, we will briefly review the definitions and concepts in the TDPCR method that are also involved in this work.

Definition 1 (Three dimensional cloud) Let $U\{(X, Y, Z) | X, Y, Z \in \mathcal{R}\}$ be a three dimensional universal, $C(E_x, En_x, He_x, E_y, En_y, He_y, E_z, En_z, He_z)$ be a qualitative concept related to U where the elements in C are individual concepts, (x, y, z) , $(x \in X, y \in Y, z \in Z)$ is a random realisation of C . Then, the distribution of (x, y, z) on U is defined as the three dimensional cloud if (x, y, z) satisfies:

$$(x, y, z) \sim N((E_x, E_y, E_z), (En'_x, En'_y, En'_z)) \quad (3)$$

$$(En'_x, En'_y, En'_z) \sim N((En_x, En_y, En_z), (He_x^2, He_y^2, He_z^2)) \quad (4)$$

$$\mu(x, y, z) = e^{-\frac{(x-E_x)^2}{2En_x'^2} - \frac{(y-E_y)^2}{2En_y'^2} - \frac{(z-E_z)^2}{2En_z'^2}} \quad (5)$$

where (En'_x, En'_y, En'_z) is a random realisation of three-dimensional normal distribution, $\mu(x, y, z)$ is the certainty degree of (x, y, z) on U .

Definition 2 (Three-dimensional piecewise cloud representation) Let T be the original time series, the three-dimensional piecewise cloud representation of T can be performed through the following three steps: (i) divide T into a series of segments $S_1, S_2, \dots, S_W$; (ii) calculate the first and second order differences of S_i , i.e., S'_i and S''_i , $(i = 1, \dots, W)$; (iii) transform each segment group $\{S_i, S'_i, S''_i\}$ into a three-dimensional cloud C_i^T through

the backward normal cloud generator (as defined in Algorithm 1).

Definition 3 (TDPCR distance) Let $\tilde{T}_1 = \{C_1^{T_1}, \dots, C_W^{T_1}\}$ and $\tilde{T}_2 = \{C_1^{T_2}, \dots, C_W^{T_2}\}$ be two time series in the form of three dimensional piecewise clouds, the distance between them is defined as

$$L(\tilde{T}_1, \tilde{T}_2) = \frac{1}{W} \sum_{i=1}^W Dist(C_i^{T_1}, C_i^{T_2}) \quad (6)$$

where $Dist(C_i^{T_1}, C_i^{T_2})$ is the distance between two clouds as defined in Si et al. [42].

4 Dynamic Sub-Sequence Warping

In this section, we describe the proposed DSSW method, which incorporates dynamic time warping with time series representation to align two time series in the feature space. The problem we aim to deal with is formally defined first, followed by detailed descriptions of the three components, namely, the time series segmentation module, the feature extraction module, and the similarity measure module. The structure of DSSW is illustrated in Fig. 1.

4.1 Problem Definition

Formally, a raw time series T is an ordered sequence of n pairs $T = ((x_1, y_1), \dots, (x_n, y_n))$ where x_i ($i = 1, \dots, n$) are the timestamps, y_i corresponds to the observation at that time. T can also be simplified to a sequence of real-valued variables $T = (t_1, \dots, t_n)$ if t_i is collected given a uniform sampling rate. However, data mining algorithms that work on the raw time series of high dimensions are computationally expensive. As a result, representing the raw time series by its fundamental features before any mining task is highly desirable to reduce processing and storage costs.

Given a time series $T = (t_1, \dots, t_n)$, a representation of T can be expressed as $\tilde{T} = (f_1, \dots, f_d)$ ($d \ll n$) where f_i ($i = 1, \dots, d$) represents scalars or vectors comprised of the key features extracted from T . After time series representation, a notion of similarity between two series in the reduced space should also be defined to support the mining tasks afterwards. Here, we specify the similarity measure between time series $\tilde{T}_1$ and $\tilde{T}_2$ as $L(\tilde{T}_1, \tilde{T}_2)$, which takes the two represented time series as input and returns the distance between them. This paper aims to propose a new similarity measure L^* . Specifically, we aim to speed up the computation of the DTW distance while lifting the restrictions of TDPCR on measuring the similarity between two represented time series of unequal length or with time shift.

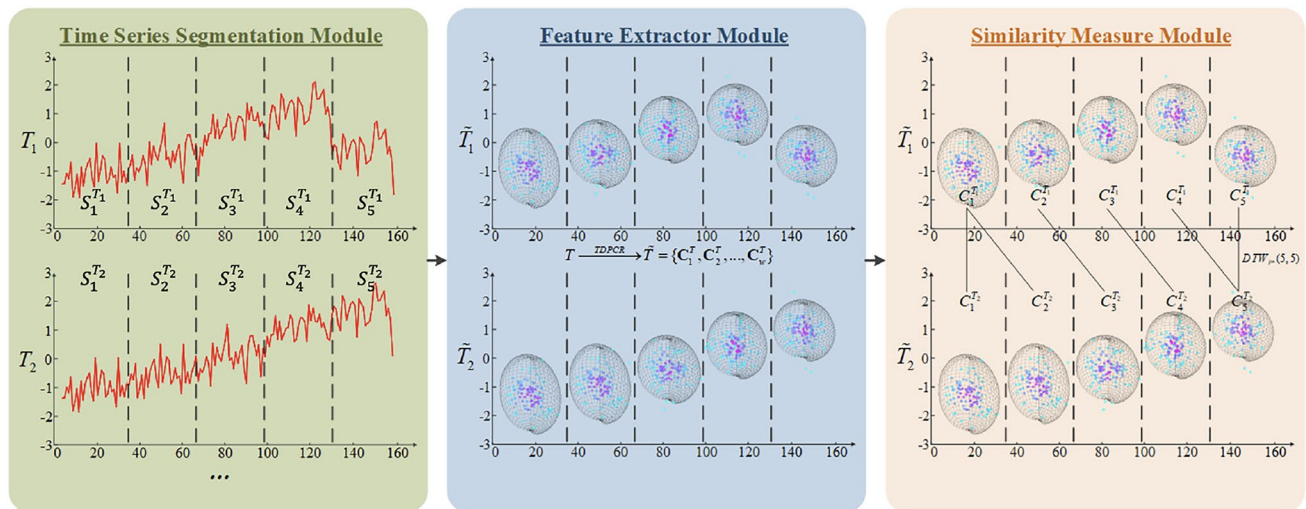


Fig. 1 The proposed DSSW method consists of a time series segmentation module, a feature extractor module, and a similarity measure module

4.2 Framework Overview

The proposed model (see Fig. 1) consists of three components: (a) A time series segmentation module, (b) A feature extractor module, and (c) A similarity measure module. First, the segmentation module divides the original time series into continuous segments by a sliding window. The three-dimensional cloud features are then extracted in the extractor module. Finally, the similarity measure module gives the similarity between two represented time series. In the following sections, we provide the details of each module.

4.3 Time Series Segmentation Module

Given a time series T of length n , the segmentation module divides the original time series into W continuous segments $T = \{S_1, \dots, S_W\}$ with the length l of each segment equaling to $\lceil n/W \rceil$ where “ $\lceil \cdot \rceil$ ” returns the rounded up number. In the original TDPCR, the authors utilised an overlap partition strategy that allows overlapping two adjacent segments to maintain the information chain and avoid the indivisible problem. However, selecting a proper value of the overlapping rate is highly problem-specific and could cause unnecessarily high processing costs. In this paper, we generate multiple segments evenly, that is, the length of segments S_i ($i = 1, \dots, W - 1$) are l while that of the last segment S_W is $n - l * (W - 1)$.

4.4 Feature Extractor Module

In the feature extractor module, the fundamental features of the original time series within each segment are extracted.

For each segment S_i , we first calculate its first-order and second-order differences, namely, S'_i and S''_i , which extract the trend and curvature information of the original time series that may be hidden after dimension reduction [42]. This results in W vectors of segments $G_i = \{S_i, S'_i, S''_i\}$ in chronological order. Then, each group is converted to a three-dimensional cloud through the backward normal cloud generator. As illustrated in Algorithm 1, The original time series is embedded into a lower dimensional space $\tilde{T} = \{C_1^T, \dots, C_W^T\}$, where C_i^T as below is a three-dimensional cloud composed of the cloud features of the i th segment:

$$C_i^T = \{E_{i,x}, En_{i,x}, He_{i,x}, E_{i,y}, En_{i,y}, He_{i,y}, E_{i,z}, En_{i,z}, He_{i,z}\}, \quad (i = 1 \dots W) \quad (7)$$

After the piecewise cloud representation, we can visualise the constructed three-dimensional cloud. Specifically, we randomly generate N samples for each segment (differentiated in colours in Fig. 1), subject to three-dimensional Gaussian distribution:

$$\mu(x_j, y_j, z_j) = e^{-\frac{(x_j - E_{i,x})^2}{2En_{i,x}^2} - \frac{(y_j - E_{i,y})^2}{2En_{i,y}^2} - \frac{(z_j - E_{i,z})^2}{2En_{i,z}^2}}, \quad (j = 1 \dots N) \quad (8)$$

$$\text{where } (En_{i,x}^*, En_{i,y}^*, En_{i,z}^*) \sim \mathcal{N}((En_{i,x}, En_{i,y}, En_{i,z}), (He_{i,x}, He_{i,y}, He_{i,z})).$$

Input: Vectors $\{X, Y, Z\}$

Output: A three dimensional cloud $C(E_x, En_x, He_x, E_y, En_y, He_y, E_z, En_z, He_z)$

1: $E_x \leftarrow \text{mean}(X)$; $E_y \leftarrow \text{mean}(Y)$; $E_z \leftarrow \text{mean}(Z)$

2: $En_x \leftarrow \sqrt{\frac{\pi}{2}} \text{mean}(|X - E_x|)$; $En_y \leftarrow \sqrt{\frac{\pi}{2}} \text{mean}(|Y - E_y|)$; $En_z \leftarrow \sqrt{\frac{\pi}{2}} \text{mean}(|Z - E_z|)$

3: $He_x = \sqrt{\text{var}(X) - En_x^2}$, $He_y = \sqrt{\text{var}(Y) - En_y^2}$, $He_z = \sqrt{\text{var}(Z) - En_z^2}$

return $C(E_x, En_x, He_x, E_y, En_y, He_y, E_z, En_z, He_z)$;

Algorithm 1 Three-dimensional backward normal cloud generator

4.5 Similarity Measure Module

A similarity measure within the embedded space is a prerequisite for further data mining tasks. We utilise a stepwise framework inline with TDPCR to compute the distance between time series $\tilde{T}_1$ and $\tilde{T}_2$, which consists of four steps: (a) Divide each three-dimensional cloud C_i^T into three one dimensional sub-clouds $C_i^T = \{C_{i,x}^T, C_{i,y}^T, C_{i,z}^T\}$; (b) Compute the overlapping area between the sub-clouds of $C_i^{T_1}$ and $C_i^{T_2}$ as the reciprocal of their distance; (c) Combine the distance of corresponding sub-clouds with optimal weights w_1 to w_3 as the distance $\text{Dist}(C_i^{T_1}, C_i^{T_2})$ between $C_i^{T_1}$ and $C_i^{T_2}$; (d) Denote the average distance of $C_i^{T_1}$ and $C_i^{T_2}$ ($i = 1, \dots, W$) as the distance between $\tilde{T}_1$ and $\tilde{T}_2$.

The original similarity measure of TDPCR works by strictly aligning two time series in the feature space. In other words, it assumes that $\tilde{T}_1$ and $\tilde{T}_2$ are comprised of the same number of three-dimensional clouds ($W_{\tilde{T}_1} = W_{\tilde{T}_2}$). However, the raw time series varies in length where TDPCR fails to be applied directly. Besides, the time shift of the characteristics or patterns of two similar raw time series will considerably lower the effectiveness of the TDPCR distance. This issue is illustrated in Fig. 1, where we create two synthetic time series of equal length but with feature shift. Specifically, segments $S_2^{T_2}$ to $S_5^{T_2}$ are generated through rightward shifting of $S_1^{T_1}$ to $S_4^{T_1}$, whereas $S_5^{T_1}$ and $S_1^{T_2}$ are derived by replicating $S_4^{T_1}$ and $S_2^{T_2}$, respectively, followed by a downward translation along the y-axis. In this case, the original TDPCR distance can be misleading as it assumes a one-to-one correspondence between cloud features ($C_i^{T_1}$ to $C_i^{T_2}$, $i = 1 \dots 5$), which leads to a large distance. To release the constraint, we incorporate dynamic time warping into the original TDPCR distance, offering the desired elasticity to align the shifted features, as shown in Fig. 1 (right).

Given two series $\tilde{T}_1 = \{C_1^{T_1}, \dots, C_{W_{\tilde{T}_1}}^{T_1}\}$ and $\tilde{T}_2 = \{C_1^{T_2}, \dots, C_{W_{\tilde{T}_2}}^{T_2}\}$, both in the form of piecewise three-dimensional cloud models, the DSSW distance of a warping path in the feature space is denoted as:

$$DSSW_P[A_s, B_s] = \sum_{i=1}^s \text{Dist}(C_{A_i}^{T_1}, C_{B_i}^{T_2})$$

$$\begin{aligned} & A_i \in [1, W_{\tilde{T}_1}] \\ & B_i \in [1, W_{\tilde{T}_2}] \\ \text{s.t. } & i \in [1, s] \\ & \max(W_{\tilde{T}_1}, W_{\tilde{T}_2}) \leq s \leq W_{\tilde{T}_1} + W_{\tilde{T}_2} - 1 \end{aligned} \quad (9)$$

where (A_i, B_i) is an index pair. Then the DSSW distance of the best warping path P^* can be obtained by:

$$DSSW_{P^*}[A_i, B_i] = \text{Dist}(C_{A_i}^{T_1}, C_{B_i}^{T_2}) + \min\{DSSW_{P^*}[A_{i-1}, B_i], DSSW_{P^*}[A_i, B_{i-1}], DSSW_{P^*}[A_{i-1}, B_{i-1}]\} \quad (10)$$

where $P^* = \arg \min_{P \in \mathcal{P}} DSSW_P[A_s, B_s]$. After the dynamic programming process, the final distance is obtained as $DSSW_{P^*}[A_s, B_s]$.

5 Empirical Analysis

This section evaluates the proposed similarity measure through a typical supervised learning task, i.e., time series classification. Experiments are conducted to show the effectiveness of DSSW from the classification accuracy and efficiency perspectives. Formally, our experiments are set up to answer the following research questions: (Q1) To what extent does the proposed DSSW outperform current DTW-based methods in terms of classification accuracy? (Q2) Is DSSW computationally efficient? (Q3) Is the in-sample performance of DSSW aligned with that in the testing phase? We will first introduce the experimental setups and then the experimental results and analysis. The code and data to reproduce the experiments are available in <https://github.com/m/zhourongleiden/DSSW>.

5.1 Experiment Setup

Benchmark Datasets We perform all the experiments on widely used benchmark datasets from the UCR time series archive [11]. Since the main objective of this work is to accelerate DTW calculation, we require relatively long time series to evaluate their performance. To this end, we follow a common routine in existing work [26, 36, 40] that utilises

a threshold (500 in this work) to select “long” time series from the archive. This will avoid cherry-picking and ensure fair and consistent comparisons. Each dataset has been originally divided into the training and the testing sets for model construction and performance evaluation, respectively. All datasets have been z-normalised to have a mean of 0 and a variance of 1.

Baseline Methods We compare the performance of DSSW with relevant DTW methods, including ESDTW [36], LE-DTW [26], 3D-DTW [40], DTW with optimal warping window length (DTW_Rn), weighted DTW (WDTW), derivative DTW with optimal warping window length (DDTW_Rn), weighted derivative (DD_DTW), and derivative transform distance (DTD_C). Among them, ESDTW, LE-DTW, and 3D-DTW are representation-based DTW methods, while ESDTW achieves state-of-the-art performance in typical time series classification tasks. Other baselines are prevalent DTW variants that improve the performance of the original DTW. We employ the One Nearest Neighbor (1-NN) as the basic classifier for all baselines since it is parameter-free and emphasises the effectiveness of the underlying distance measure [17]. Hence, a better classification performance can be purely attributed to the superiority of the corresponding similarity measure.

Evaluation Metrics The baselines and the proposed DSSW are benchmarked using the classification error rate, which is defined as the number of instances in the test set that have been classified to the wrong group divided by the size of the test set [11]. Besides, we compare the computational efficiency in terms of actual CPU execution time and the performance consistency based on the Texas Sharpshooter plot [4].

Implement Details One challenge of applying DSSW is choosing the appropriate number of segments W and the coefficients w_1 to w_3 in weighting the distance of sub-clouds pairs of two three-dimensional clouds. Since we would like to select the proper parameters during training automatically, a prominent algorithm configuration tool SMAC (sequential model-based algorithm configuration) [20] is employed for efficiently optimising these hyperparameters rather than an exhaustive grid search. For each dataset, we performed one run of SMAC v.3 to minimise the leave-one-out classification error on the training part, using a time budget of 50 steps (i.e., up to 50 parameter configurations were evaluated). The configuration space of W is limited in the range of 1 to 30 since we would like to reduce the dimension of the original time series substantially. The combining weights w_1 to w_3 are optimised within 0 to 1, satisfying $w_1 + w_2 + w_3 = 1$.

One can utilise the configuration that achieves the best in-sample classification results in testing. However, we noticed a discrepancy between model performance in the

training and testing phases due to the different distribution of sample labels. Thus, we adopted the following strategy to choose the most suitable configurations for testing. Specifically, the parameter configurations at each training step and the corresponding classification error on all training data are recorded. During testing, we identify the training samples that are most similar to the testing samples. The parameter configuration leading to the lowest classification error on these similar samples will be utilised in the testing phase.

Remark 1 We observed that the proposed method may yield suboptimal results when the configuration space of W is limited to 30, particularly for datasets containing consecutive spikes. This issue is discussed in Section 5.5, where we provide recommendations accordingly.

5.2 Comparisons on Classification Precision

In this section, we address Q1 by comparing DSSW with other baselines from the classification accuracy perspective. Table 1 shows the classification error rates of all methods on the benchmark datasets, where the best performance in each case is highlighted in bold. We also show the absolute win times of different methods and the relative win/lose/tie results of all baselines against DSSW at the end of the table. For all baselines, the results we reported are acquired from the original paper [26, 36, 40] or the literature.¹ Due to the space limit, the optimal hyperparameters of DSSW obtained during training are listed separately in Table 2. Firstly, we can see in Table 1 that DSSW and ESDTW outperform the others by a large margin with 11 and 8 win times, respectively. DTW_Rn, on the contrary, achieved the best result in neither of the 28 cases. Furthermore, comparisons in terms of win/lose/tie show that DSSW can beat other methods in most cases. Specifically, DSSW achieves lower classification error in 15 out of 28 cases than ESDTW.

To better illustrate the above results, we show the pairwise comparisons between DSSW and other methods in scatter plots, as shown in Fig. 2. In each subplot, the y-axis and x-axis represent the classification error rate given by DSSW and other baselines respectively, and each dot corresponds to a time series dataset. The dots below the diagonal indicate that DSSW performs better on that dataset. The region within the two red dashed lines ($\pm 5\%$) suggests no significant difference between the two methods. We can see from Fig. 2b–i that the vast majority of the dots are located in the white region, suggesting the better performance of DSSW over all the competitors. This is consistent with the win/lose/tie results in Table 1. Besides, quite a few of the dots are under the lower red dashed line comparing DSSW

¹ <http://www.timeseriesclassification.com/>.

Table 1 The classification error rates of all methods on the benchmark datasets

Datasets	Methods									
	DSSW	ESDTW	LE-DTW	3D-DTW	DTW_Rn	WDTW	DDTW_Rn	DD_DTW	DTD_C	DTW
BeetleFly	0.100 (2)	0.100 (2)	0.100 (2)	0.150 (4.5)	0.350 (9)	0.300 (6.5)	0.150 (4.5)	0.350 (9)	0.350 (9)	0.300 (6.5)
BirdChicken	0.050 (1)	0.100 (2.5)	0.150 (5)	0.150 (5)	0.300 (10)	0.250 (8.5)	0.100 (2.5)	0.150 (5)	0.200 (7)	0.250 (8.5)
Car	0.217 (4.5)	0.132 (1)	0.350 (9.5)	0.350 (9.5)	0.233 (8)	0.217 (4.5)	0.217 (4.5)	0.200 (2)	0.217 (4.5)	0.267 (7)
CinC_ECG_torso	0.201 (6)	0.177 (5)	0.314 (8)	0.332 (9)	0.070 (2)	0.141 (3)	0.061 (1)	0.275 (7)	0.148 (4)	0.349 (10)
Computers	0.376 (8.5)	0.296 (4)	0.264 (1)	0.416 (10)	0.376 (8.5)	0.300 (5.5)	0.324 (7)	0.284 (2.5)	0.284 (2.5)	0.300 (5.5)
Earthquakes	0.286 (6)	0.180 (1)	0.217 (2)	0.320 (9)	0.273 (3.5)	0.273 (3.5)	0.324 (10)	0.295 (7.5)	0.295 (7.5)	0.281 (5)
FordA	0.235 (1.5)	0.340 (6)	0.394 (8)	0.398 (9)	0.335 (5)	0.347 (7)	0.278 (4)	0.277 (3)	0.235 (1.5)	0.446 (10)
FordB	0.267 (1)	0.391 (8)	0.381 (7)	0.365 (5)	0.401 (9.5)	0.401 (9.5)	0.343 (3)	0.333 (2)	0.347 (4)	0.380 (6)
HandOutlines	0.196 (7.5)	0.197 (9)	0.295 (10)	0.196 (7.5)	0.122 (2)	0.130 (3)	0.189 (6)	0.132 (4)	0.135 (5)	0.119 (1)
Haptics	0.539 (1)	0.581 (3)	0.558 (2)	0.591 (5)	0.584 (4)	0.630 (9)	0.672 (10)	0.601 (6.5)	0.601 (6.5)	0.623 (8)
Herring	0.406 (2)	0.515 (10)	0.422 (3)	0.391 (1)	0.469 (8)	0.469 (8)	0.484 (9)	0.453 (4.5)	0.453 (4.5)	0.469 (8)
InlineSkate	0.476 (2)	0.573 (5)	0.593 (6)	0.738 (10)	0.613 (8)	0.596 (7)	0.544 (4)	0.438 (1)	0.491 (3)	0.616 (9)
LargeKitchenAppliances	0.472 (10)	0.158 (1)	0.197 (2)	0.235 (8)	0.205 (5)	0.205 (5)	0.243 (9)	0.205 (5)	0.205 (5)	0.205 (5)
Lighting2	0.230 (8.5)	0.180 (7)	0.098 (1.5)	0.246 (10)	0.131 (4.5)	0.098 (1.5)	0.230 (8.5)	0.131 (4.5)	0.131 (4.5)	0.131 (4.5)
MALLAT	0.090 (8)	0.112 (10)	0.082 (6)	0.100 (9)	0.086 (7)	0.062 (2)	0.075 (5)	0.051 (1)	0.073 (4)	0.066 (3)
NonInvasiveFatalECG_Thorax1	0.181 (3)	0.199 (6)	0.317 (9)	0.509 (10)	0.171 (2)	0.184 (4)	0.277 (8)	0.194 (5)	0.159 (1)	0.210 (7)
NonInvasiveFatalECG_Thorax2	0.167 (7)	0.139 (6)	0.219 (9)	0.416 (10)	0.130 (4)	0.116 (3)	0.174 (8)	0.107 (1)	0.110 (2)	0.135 (5)
OliveOil	0.100 (1)	0.12 (2)	0.133 (4)	0.233 (10)	0.133 (4)	0.167 (7.5)	0.167 (7.5)	0.167 (7.5)	0.133 (4)	0.167 (7.5)
Phoneme	0.726 (1)	0.761(6)	0.800 (9)	0.748 (4)	0.773 (8)	0.839 (10)	0.753 (5)	0.731 (2)	0.732 (3)	0.772 (7)
RefrigerationDevices	0.491 (3)	0.451 (1)	0.475 (2)	0.517 (4)	0.560 (8)	0.573 (9)	0.576 (10)	0.555 (6.5)	0.555 (6.5)	0.536 (5)
ScreenType	0.568 (6)	0.539 (2)	0.517 (1)	0.565 (5)	0.589 (8.5)	0.589 (8.5)	0.557 (3)	0.571 (7)	0.563 (4)	0.603 (10)
ShapeletSim	0.006 (1.5)	0.006 (1.5)	0.450 (9)	0.078 (3)	0.306 (5)	0.244 (4)	0.494 (10)	0.389 (7)	0.400 (8)	0.350 (6)
ShapesAll	0.122 (1)	0.170 (5)	0.27 (9)	0.313 (10)	0.198 (7)	0.188 (6)	0.155 (3)	0.150 (2)	0.162 (4)	0.232 (8)
SmallKitchenAppliances	0.267 (1)	0.272 (2)	0.347 (6)	0.355 (8)	0.328 (4)	0.325 (3)	0.331 (5)	0.360 (10)	0.352 (7)	0.357 (9)
StarLightCurves	0.031 (2)	0.007 (1)	0.055 (6)	0.095 (8)	0.102 (9)	0.105 (10)	0.036 (3)	0.038 (4.5)	0.038 (4.5)	0.093 (7)
UWaveGestureLibraryAll	0.080 (6)	0.108 (8.6)	0.125 (10)	0.091 (7)	0.038 (2)	0.034 (1)	0.073 (5)	0.065 (4)	0.062 (3)	0.108 (8.5)
Worms	0.475 (8)	0.338 (1)	0.541 (10)	0.508 (9)	0.468 (6.5)	0.468 (6.5)	0.351 (2.5)	0.416 (4.5)	0.351 (2.5)	0.416 (4.5)
WormsTwoClass	0.271 (1)	0.282 (2)	0.343 (4)	0.387 (8)	0.416 (9)	0.429 (10)	0.312 (3)	0.351 (5)	0.377 (6.5)	0.377 (6.5)
Total Wins	11	8	4	1	0	2	1	3	2	1

Table 1 (continued)

Datasets	Methods									
	DSSW	ESDTW	LE-DTW	3D-DTW	DTW_Rn	WDTW	DDTW_Rn	DD_DTW	DTD_C	DTW
Win/loss/tie	–	15/11/2	20/7/1	24/3/1	18/10/0	17/10/1	18/8/2	18/10/0	15/11/2	20/8/0
Mean rank	3.964	4.236	5.75	7.375	6.071	5.893	5.714	4.625	4.554	6.679

Table 2 The optimal hyperparameters of DSSW obtained during training

Datasets	Seg/(weights)	Datasets	Seg/(weights)
BeetleFly	26/(0.253/0.505/0.242)	MALLAT	28/(0.821/0.035/0.144)
BirdChicken	24/(0.025/0.832/0.142)	NonInvasiveFatalECG_Thorax1	13/(0.606/0.077/0.317)
Car	17/(0.256/0.677/0.067)	NonInvasiveFatalECG_Thorax2	29/(0.675/0.054/0.271)
CinC_ECG_torso	8/(0.305/0.470/0.225)	OliveOil	19/(0.142/0.489/0.369)
Computers	17/(0.256/0.677/0.067)	Phoneme	15/(0.310/0.391/0.299)
Earthquakes	1/(0.156/0.184/0.660)	RefrigerationDevices	21/(0.047/0.942/0.011)
FordA	24/(0.004/0.606/0.390)	ScreenType	28/(0.024/0.768/0.208)
FordB	27/(0.001/0.840/0.159)	ShapeletSim	23/(0.020/0.848/0.132)
HandOutlines	13/(0.406/0.536/0.057)	ShapesAll	25/(0.330/0.426/0.244)
Haptics	25/(0.017/0.903/0.080)	SmallKitchenAppliances	5/(0.623/0.069/0.308)
Herring	17/(0.176/0.310/0.514)	StarLightCurves	9/(0.621/0.054/0.326)
InlineSkate	20/(0.145/0.485/0.369)	UWaveGestureLibraryAll	24/(0.330/0.432/0.238)
LargeKitchenAppliances	22/(0.328/0.317/0.355)	Worms	18/(0.123/0.390/0.487)
Lighting2	23/(0.733/0.066/0.201)	WormsTwoClass	15/(0.088/0.484/0.428)

with LE-DTW, 3D-DTW, DTW_Rn, WDTW, and DTW, which shows the significant superiority of DSSW over these methods in terms of classification accuracy. Less dominance of DSSW is observed over ESDTW, DDTW_Rn, DD_DTW, and DTD_C, as many dots fall between the red dashed lines. However, only a few dots are above the upper red dashed line. Thus, we can conclude that DSSW outperforms current DTW-based methods regarding classification accuracy at different levels.

We further test the statistical significance of these results. First, we apply the Friedman test to check for statistically significant differences among the performance of multiple methods. Instead of using the actual classification error results, we base the comparisons on the ranks of the methods [13]. Specifically, the best-performing method on each dataset ranks 1, the second ranks 2, and so on. An average rank is given in the case of a tie. The ranking results are also shown in Table 1 within the parentheses after the actual classification error values. The Friedman test then checks whether the measured average ranks of all methods significantly differ from the mean rank of 5.5. Given a standard significance threshold of $\alpha = 0.05$, the null hypothesis, which states that all methods are comparable, is rejected with $p = 5 \times 10^{-5}$, suggesting the existence of significant differences among the performance of multiple methods. Then, we could proceed with a post-hoc Nemenyi test [14] to determine where these differences lie. It is realised by forming cliques, i.e., the methods are grouped to one *clique* if their mean ranks differ by a critical difference CD at most. Herein, the CD is given by $q_\alpha \sqrt{k(k+1)/6N}$ where q_α is

the critical value, k and N are the numbers of methods and datasets, respectively. In Fig. 3, we summarise all the comparisons in a critical difference diagram, which orders the methods based on their mean ranks. A clique of methods is indicated by a solid bar within which the grouped methods have no significant difference. We can see from Fig. 3 that DSSW ranks the highest but is statistically indistinguishable from other methods like ESDTW, DTD_C, and DD_DTW. 3D-DTW and DTW rank the lowest and are statistically inferior to other methods. The advantage of DSSW over DTW becomes clearer considering the speed-up of DSSW in the execution time, as per the discussion of question Q2 in the next section.

5.3 Comparisons on Efficiency

In this section, we address Q2 by comparing DSSW with current DTW-based methods from the computational efficiency perspective.

First, we compare the theoretical time complexity of DSSW with other baselines. The execution of DSSW-1NN consists of time series representation and similarity search in the low-dimensional feature space. As a result, the time complexity of DSSW-1NN is the sum of these two parts: (1) $O(n)$ in the piecewise cloud representation [42], and (2) $O(W^2)$ in the similarity measure where n and W is the length of the raw time series and the number of segments, respectively. This results in a total time complexity of $O(W^2)$. The computational complexity of all baselines is compared in Table 3. It can be seen that DSSW and other

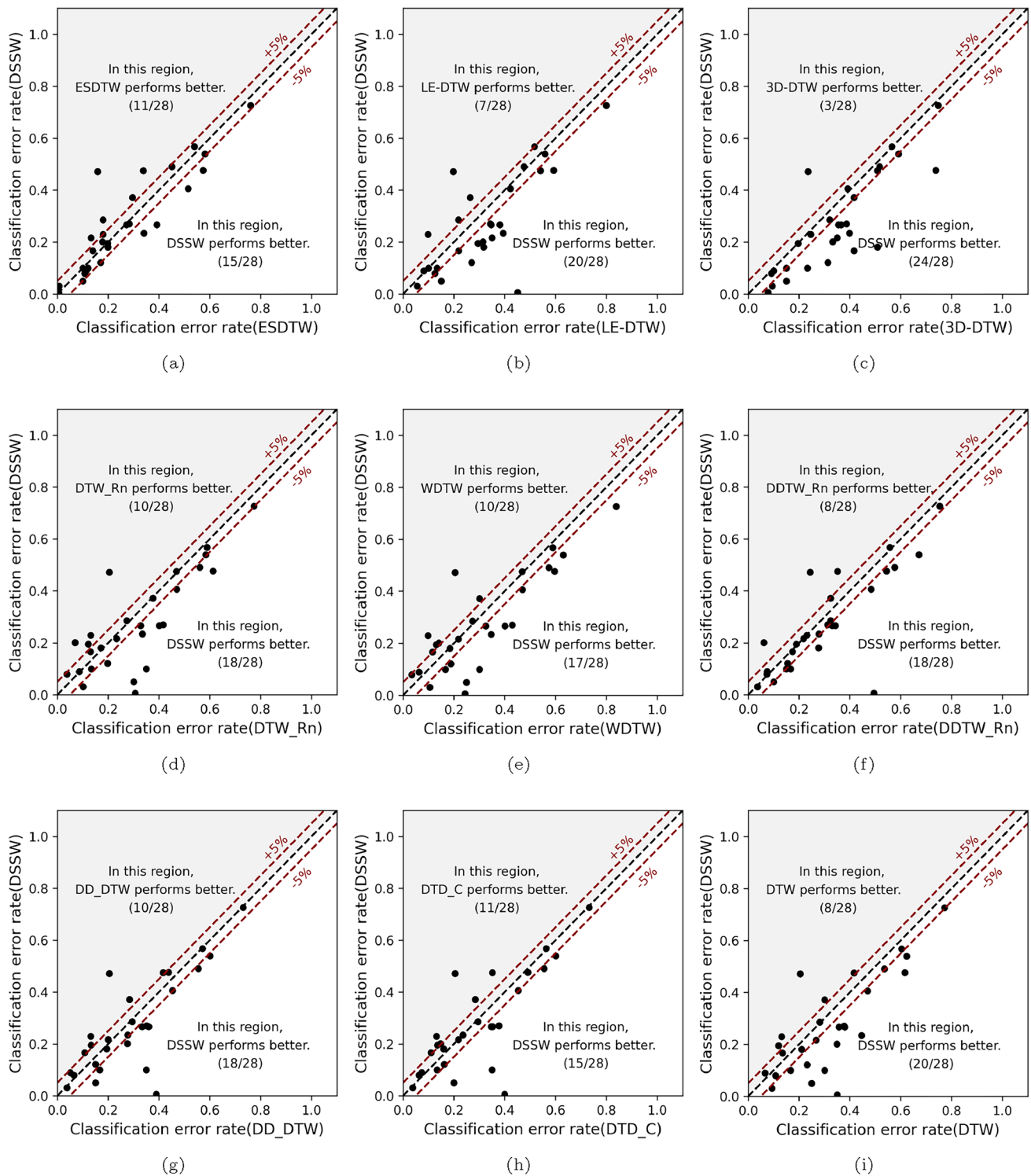


Fig. 2 Pairwise comparisons between DSSW and the other methods: **a** DSSW versus ESDTW; **b** DSSW versus LE-DTW; **c** DSSW versus 3D-DTW; **d** DSSW versus DTW_Rn; **e** DSSW versus WDTW; **f** DSSW versus DDTW_Rn; **g** DSSW versus DD_DTW; **h** DSSW ver-

sus DTD_C; **i** DSSW versus DTW. Each dot represents a time series dataset. The region within the two dashed lines suggests no significant difference between the two methods

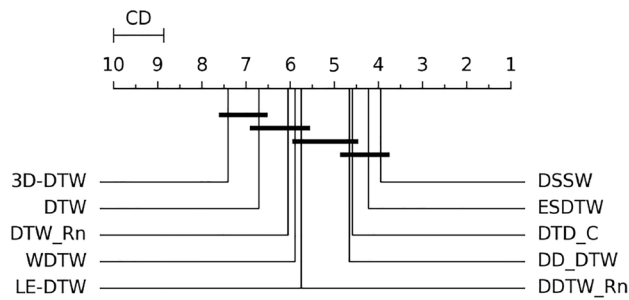


Fig. 3 Comparisons of the ranks of all methods in Table 1 in a critical difference diagram. A clique of methods is indicated by a solid bar within which the grouped methods have no significant difference (at $p = 0.05$)

Table 3 The computational complexity of different baselines

Method	Computational complexity
DSSW	$O(m^2)$
ESDTW	$O(m^2)$
LE-DTW	$O(m^2)$
3D-DTW	$O(m^2)$
DTW and other variants	$O(n^2)$

Herein, n is the length of the raw time series; m generalises the length of the time series in the reduced space, e.g., m represents the number of segments in DSSW and 3D-DTW, and the number of local extrema in ESDTW and LE-DTW, respectively. It can be assumed that $m \ll n$ holds

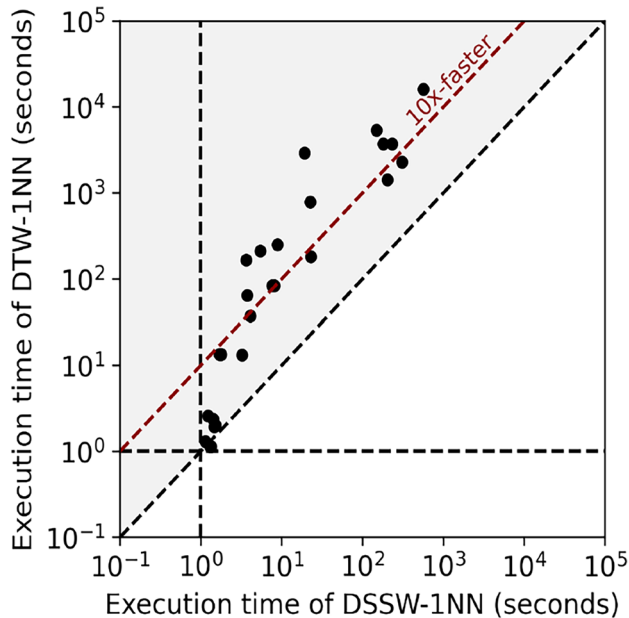


Fig. 4 The pairwise comparisons between the actual CPU execution time of DSSW-1NN and DTW-1NN on all datasets. The dots above the red dashed line indicate the significant performance leap of DSSW-1NN over DTW-1NN in terms of computation cost on that dataset

representation-based methods are substantially more efficient than the original DTW method and the variants as $m \ll n$.

Furthermore, we show the pairwise comparisons between the actual CPU execution time of DSSW-1NN and DTW-1NN on all datasets in Fig. 4. Both methods are tested on an Intel i7-10700K (3.8 GHz) machine. The implementation of DTW for both methods is acquired from tslearn.² Due to the broadly varied range of values, we present the execution time values of DSSW-1NN and DTW-1NN on a logarithmic scale on the y-axis and x-axis, respectively. We can see in the figure that most of the dots (26 out of 28 cases) fall in the grey region where DSSW-1NN is faster than DTW-1NN. Specifically, there are 14 out of 28 cases where DSSW-1NN is at least an order of magnitude faster (10x-faster) than DTW-1NN. This shows the huge performance leap of DSSW-1NN over DTW-1NN in computation cost.

We also notice that DTW-1NN is more efficient on the BeetleFly and BirdChicken datasets. Table 4 shows the details of the two datasets along with the comparisons between DSSW-1NN and DTW-1NN on these datasets in terms of classification error and computation cost. We note that the sizes of these datasets are much smaller than others. This leads to a lower execution time of DTW-1NN with quadratic time complexity. However, the differences in time are all within seconds. DSSW performs reasonably better, considering the trade-offs between the execution time and the classification error.

5.4 Comparisons on Performance Consistency

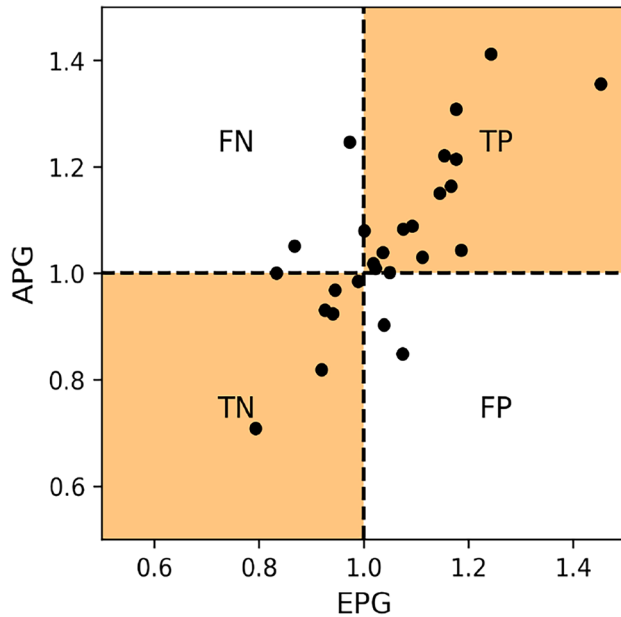
We have demonstrated the effectiveness of DSSW from the classification accuracy and efficiency perspectives by addressing Q1 and Q2. On the other hand, the performance consistency of a method in the training and testing phases is also important if we want to apply it to a new problem when the testing part has not been revealed yet.

In this section, we further demonstrate the superiority of DSSW over DTW by comparing their performance consistency in classification accuracy, as per the research question Q3. Specifically, we are interested in predicting in which cases DSSW-1NN will have higher classification accuracy. This is achieved by constructing the Texas Sharpshooter plot, which illustrates whether the ratios of the in-sample classification errors of the two methods align with the ratios of their classification errors in the testing phase. To create the plot, we first calculate the performance gain, that is, the classification accuracy ratio of DSSW-1NN and DTW-1NN. The Texas Sharpshooter plot compares the expected performance gain (EPG) versus the actual performance gain

² <https://tslearn.readthedocs.io/>

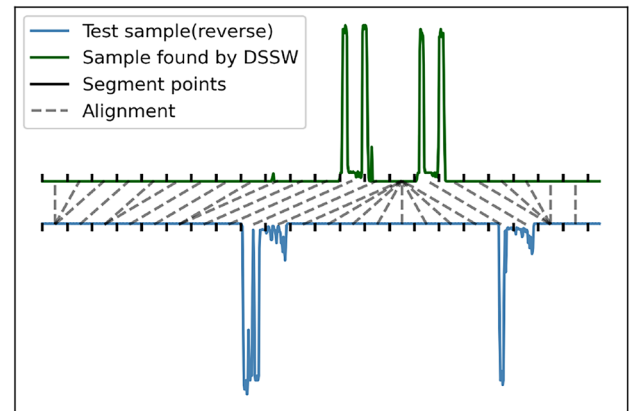
Table 4 The detailed comparisons between DSSW-1NN and DTW-1NN on the BeetleFly and BirdChicken datasets in terms of classification error and computation cost

Datasets	Training size	Testing size	Length	Execution time		Classification error	
				DSSW-1NN	DTW-1NN	DSSW-1NN	DTW-1NN
BeetleFly	20	20	512	1.308	1.145	0.100	0.300
BirdChicken	20	20	512	1.341	1.139	0.050	0.250

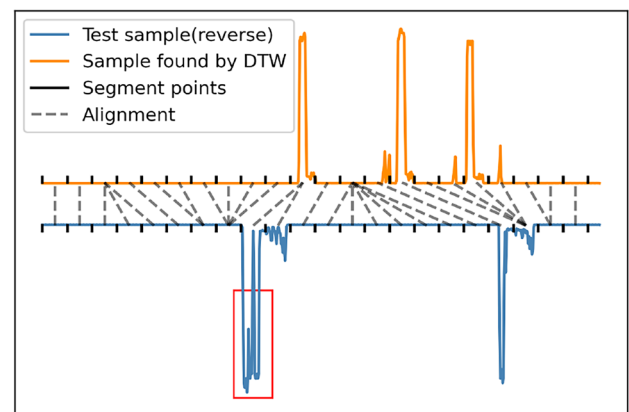
**Fig. 5** Comparisons of DSSW-1NN and DTW-1NN on all datasets via Texas Sharpshooter plot where EPG and APG represent the expected and actual performance gain, respectively

(APG) of the two methods. Herein, EPG is calculated based on the training part, while APG is calculated based on the testing part. If the value of EPG is greater than 1, we expect that DSSW-1NN could achieve higher classification accuracy in the testing phase with an APG greater than 1. As presented in Fig. 5, each dot falls into one of the four quadrants: (i) True positive (TP). In this region, the EPG and APG are greater than 1, meaning DSSW-1NN outperforms DTW-1NN in both the training and testing phases. (ii) True negative (TN). In this region, the EPG and APG are lower than 1, which means that DTW-1NN performs better in both the training and testing phases. (iii) False negative (FN). In this region, DSSW-1NN works better in the testing phase even with an EPG lower than 1. (iv) False positive (FP). In this region, we expect that DSSW-1NN outperforms DTW-1NN in the testing phase, given an EPG greater than 1, but DSSW-1NN does worse.

Ideally, all points in the plot should be in either TP or TN so that the performance in the training phase is consistent with that in the testing phase. We can see in Fig. 5 that the vast majority of dots fall into these two regions. Only 2 out of 28 dots fall into the FP, where we wrongly expect DSSW to perform better. As a result, we can conclude that the



(a) The most similar sample found by DSSW



(b) The most similar sample found by DTW

Fig. 6 Visualisation of the similarity measure process of DSSW between samples from the *LargeKitchenAppliances* dataset

performance of DSSW-1NN is consistent and better than that of DTW-1NN in most cases regarding the classification of long time series.

5.5 Discussions

As demonstrated in previous experiments, DSSW achieved higher classification accuracy while reducing computational costs compared to existing DTW-based methods. However, we observed suboptimal performance of DSSW in certain scenarios. In this section, we explore the limitations of the proposed method by visualising the similarity measure results on samples from the *LargeKitchenAppliances*

dataset, where DSSW demonstrates less favourable outcomes. This analysis aims to identify areas for improvement and enhance the applicability of DSSW to a broader range of cases.

In Fig. 6a, the most similar sample found by DSSW from the whole training set to a sample from the testing set are viewed in green and blue, respectively. Both samples are divided into segments and transformed into three-dimensional cloud representations based on the optimal hyperparameters of DSSW obtained during training. The best warping path constructed based on Eq. 10 is visualised via dashed lines. We can see that the segments containing spikes are aligned well, resulting in a relatively low DSSW distance between the two samples. Nevertheless, the similar sample found by DSSW, which contains 4 spikes, is from a different class than the test sample, which contains 3 spikes. The main reason is that there is more than one spike in a segment, but this information is not retained after the cloud representation. To further explain the issue, we pair the test sample with the most similar sample found by DTW that belongs to the same class in Fig. 6b. Herein, both samples are turned into cloud representations based on the same hyperparameters used in Fig. 6a. The DSSW distance between the two samples is relatively large because the first two spikes in the rectangular area are wrongly merged after representation, leading to a mismatch with the rightmost spike. Similar issues can be found in other datasets that involve consecutive spikes, such as *Computers* and *Lightning2*. As a result, it is recommended to extend the configuration space of W when dealing with this type of data, such that spikes can be assigned to different segments. To demonstrate this, we conducted additional experiments using the *LargeKitchenAppliances* dataset as an example. Specifically, we expanded the configuration space of W to 1–100 to enable finer-grained segmentation, and increased the time budget to 100 steps to allow SMAC to more thoroughly explore the space. We also set the combining weights w_1 to w_3 to 1/3 to neutralise the influence of other hyperparameters. The results show that the classification accuracy decreased from 0.472 to 0.211 when using the optimised value of $W = 60$ identified by SMAC. In practice, the upper bound of the configuration space for W can be chosen to be relatively large (e.g., 10% of the time-series length) to ensure that the optimised value of W can be found. However, this also incurs additional training time, as SMAC requires more evaluations to explore the expanded search space.

So far we have addressed research questions Q1–Q3 and analysed the suboptimal performance of DSSW affected by the size of the configuration space of W . In the following, we further analyse its sensitivity to the choice of W , which addresses Q4. Specifically, we allow W to vary across the full configuration space, and report the corresponding

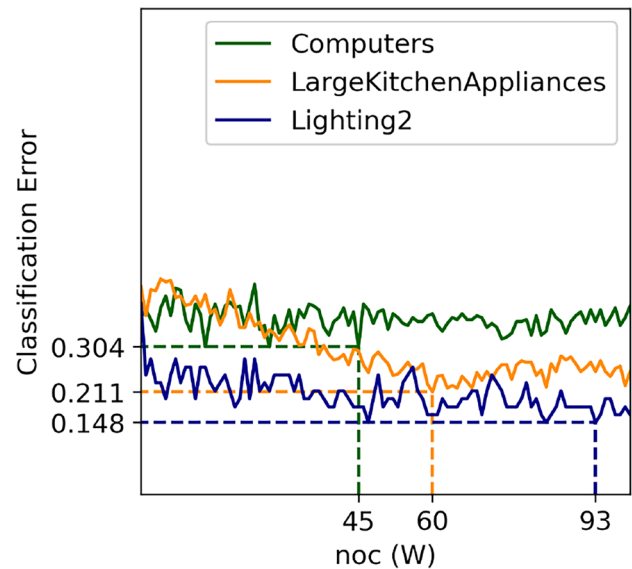


Fig. 7 Sensitivity analysis of DSSW to the number of segments W

classification accuracy. The combining weights w_1 to w_3 have been set to 1/3 by default. In Fig. 7, we present the classification accuracy on the three datasets containing consecutive spikes with respect to different values of W . Overall, the performance of DSSW remains stable across different values of W , with classification accuracy generally improving as W increases due to the retention of more information. This demonstrates the robustness of the proposed method with respect to the number of segments W . In addition, the optimal value of W , indicated by the dotted lines, can be identified by SMAC when provided with an expanded configuration space and sufficient time budget.

6 Conclusions

This paper proposes DSSW, a new time series similarity measure based on dynamic time warping and three-dimensional piecewise cloud representation. Instead of operating on raw time series, DSSW first applies dimensionality reduction via data representation and then computes the DTW distance in the reduced-dimensional feature space. The proposed DSSW method is tested and compared with prevalent DTW-based methods on time series classification tasks. The experimental results demonstrate the superiority of DSSW from three aspects: (1) DSSW outperforms the prevalent DTW-based methods at different levels in terms of classification accuracy; (2) DSSW achieves a huge performance leap over DTW concerning the computation efficiency in the classification of long time series; (3) DSSW shows consistent performance in the training and the testing phases. However, DSSW may exhibit suboptimal performance if the number of segments is not optimised, especially when

dealing with consecutive spikes. In future works, we will investigate methods to adaptively divide time series data based on the trend rather than generate segments evenly.

Acknowledgements The authors would like to thank Prof. Eamonn Keogh and everyone who contributed to the UCR time series classification archive.

Author Contributions Zhou Zhou: Manuscript writing, Methodology, Experiment; Gang Huang: Source, Revision; Laura Dawkins: Validation, Revision.

Funding This work is supported by the Youth Talent Program of Sci-Tech Think Tank under Grant XMSB20240710087 and the National Key Research and Development Program of China under Grant 2023YFB3107603.

Availability of data and materials The code and data to reproduce the experiments are available in <https://github.com/zhourongleiden/DSSW>.

Declarations

Conflict of interest The authors declare that they have no Conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Abanda A, Mori U, Lozano JA (2019) A review on distance based time series classification. *Data Min Knowl Disc* 33(2):378–412
- Agrawal R, Faloutsos C, Swami A (1993) Efficient similarity search in sequence databases. In: *International conference on foundations of data organization and algorithms*, pp 69–84
- Bai B, Li G, Wang S, Wu Z, Yan W (2021) Time series classification based on multi-feature dictionary representation and ensemble learning. *Expert Syst Appl* 169:114162
- Batista GE, Keogh EJ, Tatar OM, de Souza V (2014) CID: an efficient complexity-invariant distance for time series. *Data Min Knowl Disc* 28(3):634–669
- Berndt DJ, Clifford J (1994) Using dynamic time warping to find patterns in time series. *KDD Workshop* 10:359–370
- Borchmann L, Jurkiewicz D, Graliński F, Górecki T (2021) Dynamic boundary time warping for sub-sequence matching with few examples. *Expert Syst Appl* 169:114344
- Cai Q, Chen L, Sun J (2015) Piecewise statistic approximation based similarity measure for time series. *Knowl Based Syst* 85:181–195
- Carmona-Poyato Á, Fernández-García NL, Madrid-Cuevas FJ, Durán-Rosal AM (2021) A new approach for optimal offline time-series segmentation with error bound guarantee. *Pattern Recognit* 115:107917
- Chan K-P, Fu AW-C (1999) Efficient time series matching by wavelets. In: *Proceedings 15th international conference on data engineering* (cat. no. 99cb36337), pp 126–133
- Chen L, Özsu MT, Oria V (2005) Robust and fast similarity search for moving object trajectories. In: *Proceedings of the 2005 ACM SIGMOD international conference on management of data*, pp 491–502
- Chen Y, Keogh E, Hu B, Begum N, Bagnall A, Mueen A, Batista G (2015) The UCR time series classification archive. http://www.cs.ucr.edu/~eamonn/time_series_data/
- Das G, Gunopulos D, Mannila H (1997) Finding similar time series. In: *European symposium on principles of data mining and knowledge discovery*, pp 88–100
- Dau HA, Bagnall A, Kamgar K, Yeh C-CM, Zhu Y, Gharghabi S, Keogh E (2019) The UCR time series archive. *IEEE CAA J Autom Sin* 6(6):1293–1305
- Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. *J Mach Learn Res* 7:1–30
- Deng W, Wang G, Xu J (2016) Piecewise two-dimensional normal cloud representation for time-series data mining. *Inf Sci* 374:32–50
- Esling P, Agon C (2012) Time-series data mining. *ACM Comput Surv* 45(1):1–34
- Geler Z, Kurbalija V, Ivanović M, Radovanović M (2022) Elastic distances for time-series classification: Itakura versus Sakoe–Chiba constraints. *Knowl Inf Syst* 64(10):2797–2832
- Herrmann M, Webb GI (2021) Early abandoning and pruning for elastic distances including dynamic time warping. *Data Min Knowl Discov* 35(6):2577–2601
- Herrmann M, Webb GI (2023) Amercing: an intuitive and effective constraint for dynamic time warping. *Pattern Recognit* 137:109333
- Hutter F, Hoos HH, Leyton-Brown K (2011) Sequential model-based optimization for general algorithm configuration. In: *International conference on learning and intelligent optimization*, pp 507–523
- Imamura M, Nakamura T (2024) Efficient discovery of time series motifs under both length differences and warping. In: *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pp 1188–1198
- Itakura F (1975) Minimum prediction residual principle applied to speech recognition. *IEEE Trans Acoust Speech Signal Process* 23(1):67–72
- Keogh E, Chakrabarti K, Pazzani M, Mehrotra S (2001) Locally adaptive dimensionality reduction for indexing large time series databases. In: *Proceedings of the 2001 ACM SIGMOD international conference on management of data*, pp 151–162
- Keogh E, Ratanamahatana CA (2005) Exact indexing of dynamic time warping. *Knowl Inf Syst* 7:358–386
- Keogh EJ, Pazzani MJ (2000) Scaling up dynamic time warping for datamining applications. In: *Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining*, pp 285–289
- Lahreche A, Boucheham B (2021) A fast and accurate similarity measure for long time series classification based on local extrema and dynamic time warping. *Expert Syst Appl* 168:114374
- Li D (1997) Knowledge representation in KDD based on linguistic atoms. *J Comput Sci Technol* 12(6):481–496
- Li H (2021) Time works well: dynamic time warping based on time weighting for time series data mining. *Inf Sci* 547:592–608

29. Li J, Izakian H, Pedrycz W, Jamal I (2021) Clustering-based anomaly detection in multivariate time series data. *Appl Soft Comput* 100:106919
30. Li X, Xu S, Yang Y, Lin T, Mba D, Li C (2024) Spherical-dynamic time warping—a new method for similarity-based remaining useful life prediction. *Expert Syst Appl* 238:121913
31. Lin J, Keogh E, Wei L, Lonardi S (2007) Experiencing sax: a novel symbolic representation of time series. *Data Min Knowl Discov* 15(2):107–144
32. Liu Y-T, Zhang Y-A, Zeng M (2019) Adaptive global time sequence averaging method using dynamic time warping. *IEEE Trans Signal Process* 67(8):2129–2142
33. Maharaj EA, D’Urso P, Caiado J (2019) Time series clustering and classification. Chapman and Hall/CRC
34. Menon S, Addula SR, Parkavi A, Subbalakshmi C, Dhandayuthapani VB, Pokkuluri KS, Soni A (2024) Streamlining task planning systems for improved enactment in contemporary computing surroundings. *SN Comput Sci* 5(8):993
35. Pu F, Xu J (2025) TSQ: an optimized framework for efficiently answering time series queries. *Data Sci Eng* 6:1–18
36. Qiu L, Qiu C, Song C (2024) ESDTW: extrema-based shape dynamic time warping. *Expert Syst Appl* 239:122432
37. Ratanamahatana CA, Keogh E (2005) Three myths about dynamic time warping data mining. In: *Proceedings of the 2005 SIAM international conference on data mining*, pp 506–510
38. Sakoe H, Chiba S (1978) Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans Acoust Speech Signal Process* 26(1):43–49
39. Savaglio C, Mazzei P, Fortino G (2024) Edge intelligence for industrial IoT: opportunities and limitations. *Procedia Comput Sci* 232:397–405
40. Sharabiani A, Darabi H, Rezaei A, Harford S, Johnson H, Karim F (2017) Efficient classification of long time series by 3-D dynamic time warping. *IEEE Trans Syst Man Cybern Syst* 47(10):2688–2703
41. Shen DS, Chi M (2022) TC-DTW: accelerating multivariate dynamic time warping through triangle inequality and point clustering. *Inf Sci* 621:611–626
42. Si G, Zheng K, Zhou Z, Pan C, Xu X, Qu K, Zhang Y (2018) Three-dimensional piecewise cloud representation for time series data mining. *Neurocomputing* 316:78–94
43. Vijay RK, Nanda SJ (2023) Earthquake pattern analysis using subsequence time series clustering. *Pattern Anal Appl* 26(1):19–37
44. Webb GI, Petitjean F (2021) Tight lower bounds for dynamic time warping. *Pattern Recognit* 115:107895
45. Wu R, Keogh EJ (2020) FastDTW is approximate and generally slower than the algorithm it approximates. *IEEE Trans Knowl Data Eng* 34(8):3779–3785
46. Yu M, Huang H, Hou R, Ma X, Yuan S (2024) A deep graph kernel-based time series classification algorithm. *Pattern Anal Appl* 27(3):73
47. Zhan P, Sun C, Hu Y, Luo W, Zheng J, Li X (2020) Feature-based online representation algorithm for streaming time series similarity search. *Int J Pattern Recognit Artif Intell* 34(05):2050010
48. Zhou Z, Baratchi M, Si G, Hoos HH, Huang G (2023) Adaptive error bounded piecewise linear approximation for time-series representation. *Eng Appl Artif Intell* 126:106892