Knowl Inf Syst (2018) 57:545–570
https://doi.org/10.1007/s10115-017-1139-9

REGULAR PAPER

Sequential data classification by dynamic state warping

Zhichen Gong[1] · Huanhuan Chen[1]

Received: 17 May 2017 / Revised: 11 October 2017 / Accepted: 14 November 2017 /
Published online: 29 November 2017

Abstract The ubiquity of sequences in many domains enhances significant recent interest in sequence learning, for which a basic problem is how to measure the distance between sequences. Dynamic time warping (DTW) aligns two sequences by nonlinear local warping and returns a distance value. DTW shows superior ability in many applications, e.g. video, image, etc. However, in DTW, two points are paired essentially based on point-to-point comparisons without considering the autocorrelation of sequences. Thus, points with different semantic meanings, e.g. peaks and valleys, may be matched providing their coordinate values are similar. As a result, DTW may be sensitive to noise and poorly interpretable. This paper proposes an improved alignment method, dynamic state warping (DSW). DSW integrates the dynamic information of sequences into DTW by converting each time point into a latent state. Alignment is performed by using the state sequences. Thus, DSW is able to yield alignment that is semantically more interpretable than that of DTW. Using one nearest neighbour classifier, DSW shows significant improvement on classification accuracy in comparison with Euclidean distance (68/85 wins), DTW (70/85 wins) and its variants. We also empirically demonstrate that DSW is more robust and scales better to long sequences than Euclidean distance and DTW.

Keywords Representation learning · Reservoir computing · Time series classification

1 Introduction

Sequences are generated and analysed in almost every domain of human society such as patient treatment [1], medical [2], engineering [3], entertainment [4], etc. Computing the

✉ Huanhuan Chen
hchen@ustc.edu.cn

Zhichen Gong
zcgong@mail.ustc.edu.cn

[1] School of Computer Science and Technology, University of Science and Technology of China, Hefei, China

distance between sequences is critical to classification and has attracted significant research interest [5–8].

For sequence classification, one nearest neighbour (1NN) classifier has been empirically shown to be a strong solution with proper distance measurements [9–11]. 1NN classifier is intrinsically parameter-free. In this case, the only concern is how to measure the distance between sequences properly [12].

Note that sequences are different from typical vectorial data. Sequences are high dimensional, autocorrelated among temporal axes and possibly of varying length. Therefore, to measure the distance between sequences, consideration has to be given to the properties of sequences, such as nonlinear local warping, phase shift and scaling distortion, etc. A more comprehensive review can be found in [12].

Dynamic time warping (DTW) [13,14] is to compute the distance between two sequences by warping them locally to the same length. It allows one-to-many mappings between sequences to "stretch" a sequence or many-to-one mappings to "condense" a sequence. In this way, DTW is naturally compatible with phase distortion invariance of sequential data [12]. Despite the simplicity of DTW, 1NN classifier with DTW distance has been very successful in many applications, such as video, image and audio. [11,15,16]. It has been a consensus that DTW may be the strongest distance measurements for sequences [17,18].

DTW aligns two sequences by taking the point-to-point comparison of coordinate values as a fundamental unit [18]. Concretely, one point on a sequence is compared to all or a subset (also known as the warping window [11]) of points on the other sequence to compute a point-level distance. In doing so, one can find an alignment path such that the aligned sequences yield a globally minimum distance [13], which is returned as the distance between the original sequences. However, this point-level comparison usually cannot provide dynamic evidence for matching two points. Note that from a human's intuition, we intend to pair two points of two sequences by taking into account the nearby points or even the global structure. Thus, DTW does align globally but is unable to take the autocorrelated structure information into consideration properly [10]. This could make DTW brittle to noise [14,19], which is also demonstrated in our experiment (Sect. 4). As pointed out in [10,20], DTW usually has weak interpretability of its alignment. Concretely, the alignment result may lack local semantic meanings. For example, DTW may match points on a local peak and a valley if their Euclidean distance (or other p-norm distances) is small. Besides, DTW may match one point to too many points resulting in un-intuitive alignment results and degrade the classification performance [3]. This problem can be alleviated by constraining the possible alignments. However, "correct" alignment may be prevented from being found.

Figure 1 illustrates the DTW match result of the GunPoint dataset from UCR time series archive [21]. The two sequences are from two different classes. Briefly speaking, GunPoint dataset contains two classes of motion track sequences behaved by both an actress and an actor. Each motion includes three processes: raising a hand, pointing and putting down the hand. For the first class, there is a gun in the hand. For the second class, there is no gun. In time points roughly ranging in [20–40] and [120–140], the two sequences differ slightly by whether the actress/actor takes a gun or not. According to Fig. 1, DTW fails to capture this difference, and it returns nearly "perfect" alignment for two semantically different sequences. Besides, it is shown that DTW un-intuitively maps one point to many points on the other sequence.[1]

[1] Since our concern is the effectiveness of incorporating the dynamics into time point representation instead of the parameter tuning procedure, this motivates us to compare DSW and DTW both without constraining the size of the warping window parameter.

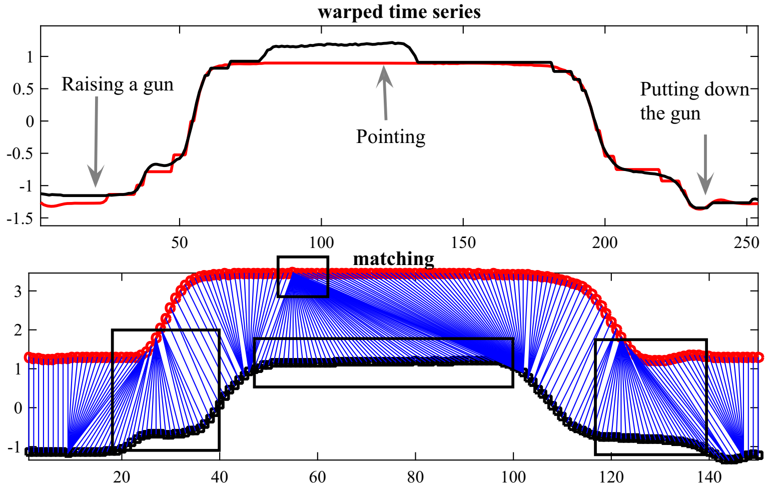


Fig. 1 DTW alignment of the GunPoint dataset. The sequences present three stages clearly, i.e. raising a hand, pointing and putting down the hand. DTW is unable to find the discriminative features between two sequences ([20–40] and [120–140]). We also observe a severe distortion that DTW matches one point on the upside sequence to almost the whole subsequence of the downside sequence

Time points of a sequence have dependencies over the time axis, which provides latent regime characterizing the dynamic behaviour. DTW is unable to take into consideration this autocorrelation information. To mitigate the problem with DTW and motivated by the advance of representation learning, in this paper, we propose an improved sequence alignment algorithm, dynamic state warping (DSW). DSW provides a simple but flexible solution for the above problem. DSW efficiently converts time points on a sequence into the corresponding hidden states, which has integrated characteristics of the past history and the current point. In this way, the state evolving sequence may be "aware" of the sequential order information and encodes the generating mechanism of original sequences. A dynamic programming technique is employed to align the state sequences instead of the temporal points. Therefore, DSW is prone to match time points with similar states together.

The time complexity of the state converting process for two given sequences of length L_Q and L_C is $O(L_Q + L_C)$. The alignment process is of $O(L_Q L_C)$[2] time complexity. Then, the overall cost is $O(L_Q L_C)$, at the same level as that of DTW.

Our method has several advantages: (1) DSW explicitly takes advantages of the autocorrelation structure characteristics of sequential data. It provides versatile and flexible discriminative state representations for sequences. (2) The alignment result of DSW is semantically more interpretable than plain DTW. (3) Using one nearest neighbour classifier, DSW exhibits lower error rates than DTW on most datasets. (4) The flexibility of DSW allows users to fine-tune the representations of sequences to capture the discriminative features for specific tasks. (5) DSW is able to deal with univariate and multivariate sequences without adjusting the algorithm. (6) DSW shows more robustness to noise and more suitable for long sequences than compared methods. (7) After obtaining the state sequences, the alignment is

[2] By employing constraints on the alignment, the complexity of DTW alignment could be linear [11]. In this paper, we focus on the effectiveness of distance measurement based on states of time points, instead of the original time points. To ease the comparison, we will not consider constraint strategy. But it is straightforward to incorporate alignment constraints in DSW.

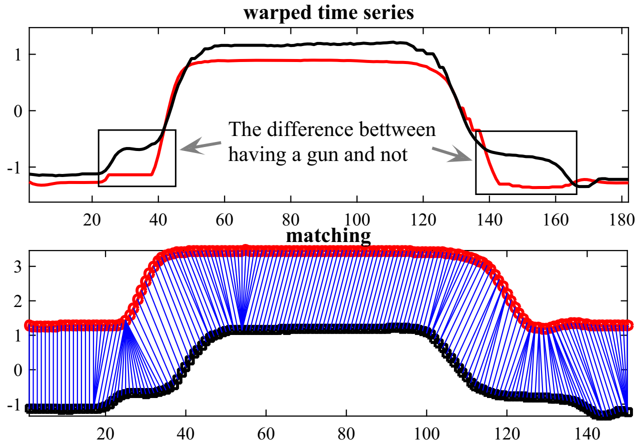


Fig. 2 DSW alignment of the GunPoint dataset. The alignment of DSW is semantically more sensible than that of DTW. The discriminative features, i.e. having a gun or not, are detected

performed by DTW; thus, advanced techniques [11,22] for improving the effectiveness and efficiency of DTW are also compatible with DSW.

As a comparison, Fig. 2 demonstrates the alignment result of DSW on GunPoint dataset. The parameters of DSW are determined randomly. It is noticeable that DSW is able to match two sequences correctly and consistently with the three true actions. The result demonstrates that DSW is beneficial to reduce the sensitiveness of warping window parameter.

Our contributions in this work include: (1) we propose to integrate the dynamic information into the DTW. This algorithm provides more interpretable and accurate alignments by taking advantage of the dynamic information of sequences. (2) By combining with one nearest neighbour classifier, our method achieves significantly better classification results than that of DTW. (3) We perform extensive experiments comparing DSW with Euclidean distance (ED) and DTW. We also empirically analyse the related properties of the state converting component of DSW extensively. (4) Our strategy provides a general framework and could incorporate other temporal filters into DTW. In addition, it is possible to incorporate discriminative learning and representation learning techniques.

The rest of this paper is organized as follows: in Sect. 2, we introduce preliminary knowledge about time series classification, DTW and related work about reservoir computing; in Sect. 3, we introduce DSW in detail; Sect. 4 performs extensive experiments to evaluate DSW; finally, Sect. 5 concludes this paper.

2 Background and related work

In this section, we first clarify some notations and basic knowledge about sequence learning. Then, we introduce the DTW algorithm in detail. Since our method is based on reservoir computing, we introduce reservoir computing at last.

2.1 Sequence distance

A sequence is a series of observations for at least one variable. We use a matrix $X = [x^1, x^2, \ldots, x^{L_X}]^T \in R^{L_X \times d}$ to denote a sequence, where $x^i \in R^{d \times 1} (i \in [1, L_X])$ is an

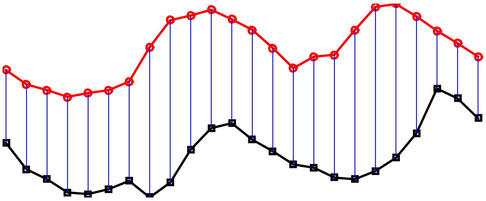


Fig. 3 Illustration of Euclidean distance

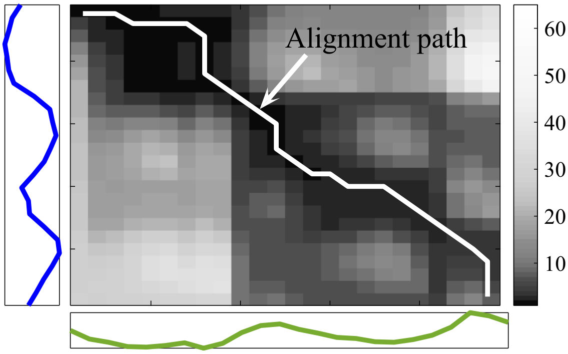


Fig. 4 Illustration of searching for the optimal alignment using DTW. The two sequences are demonstrated on the left and bottom. The pointwise distance matrix and the optimal alignment path are presented

observation at a time point indexed by i, L_X is the length of series X and d is the number of variables. Each sequence is associated with a label y. The task of classification is to learn a function mapping from X to y.

Given two sequences Q and C, if they are of equal length, one can easily compute their distance using Euclidean distance (ED) or p-norm distance [23], such as $\mathrm{ED}(Q, C) = \sqrt{\sum_{i=1}^{L}\sum_{j=1}^{d}((q^{i,j} - c^{i,j})^2)}$. This kind of distance is also called lockstep distance since it matches elements of sequences according to their positions (see Fig. 3).

However, in most real-world applications, sequences may be of varying length. In this case, elastic distance, such as DTW and longest common subsequence [17], is used to calculate the distance.

2.1.1 Dynamic time warping

Given two sequences Q and C of possibly different lengths, DTW stretches or condenses two sequences to the same length. DTW allows nonlinear local warping to take into account possible distortions. It finds an optimal alignment between Q and C such that the accumulative Euclidean distance is minimized [14]. In doing so, it returns a distance value that measures the similarity between the two sequences.

Denote $D \in R^{L_Q \times L_C}$ as the distance matrix of time points of Q and C. DTW algorithm travels from the position $[1, 1]$ to position $[L_Q, L_C]$ of D to find a sequence of indices A and B of the same length $l \geq \max\{L_Q, L_C\}$ such that the cumulative distance $\sum_{i=1}^{l} D_{A_i,B_i}$ is minimized. In this way, the point $A(i)$ of sequence Q is matched with the point $B(i)$ of sequence C. See Fig. 4 for an illustration.

To be a valid alignment, the paths A and B have to satisfy three constraints:

1. $A(1) = 1, B(1) = 1$
2. $A(l) = L_Q, B(l) = L_C$
3. $\forall i \in [1, l-1], (A(i+1), B(i+1)) - (A(i), B(i)) = \{(0, 1), (1, 0), (1, 1)\}$

Hence, the alignment of two sequences should start from the first time point and end with the last point. Each time point has at least one matching point on the other sequence. The match increases monotonically.

Formally, denote $\alpha \in \{0,1\}^{l \times L_Q}$ and $\beta \in \{0,1\}^{l \times L_C}$ be two warping matrices corresponding to the warping path A and B. Let $\alpha(i, A(i)) = 1, \beta(i, B(i)) = 1$, where $i \in \{1, 2, \ldots, l\}$. All other entries are zero. In this case, the two warped sequences become $\alpha \times Q$ and $\beta \times C$. The overall cost function of DTW is generalized as:

$$\text{Cost} = \arg\min_{l,A,B} (\sum_{i=1}^{l} D_{A_i,B_i}) = \arg\min_{l,\alpha,\beta} \sqrt[p]{(\alpha \times Q - \beta \times C)^p}. \tag{1}$$

Therefore, DTW is essentially p-norm distance [24] except that local distortion is allowed. The above cost function can be solved with time complexity $O(L_Q L_C)$ using dynamic programming:

$$\text{Cost}(m,n) = D(m,n) + \min \left\{ \begin{array}{l} D(m-1,n), \\ D(m-1,n-1), \\ D(m,n-1) \end{array} \right\}$$

2.1.2 Related work

Numerous trials have been made to enhance DTW, which can be roughly divided into two categories: adjusting the pointwise distance (e.g. [14,15,25]) and heuristically constraining the DTW (e.g. [12,22]).

Garreau et al. [25] propose to learn a distance metric for measuring the similarity between two points. This method takes the difference between the empirical alignment and true alignment as the cost function. However, the true alignments among sequences are not available for many real-world applications. Besides, it is only feasible for multivariate sequence alignment. Zhou et al. [15] combined DTW with canonical correlation analysis (CCA), termed canonical time warping (CTA). The pointwise distance is linearly transformed by CCA. CTA alternatively optimizes the CCA and DTW alignment to minimize the accumulative distance. However, CTA is only applicable for multivariate sequences. The objective function of CTW is non-convex and may return local minima.

In [26], Petitjean et al. propose to replace multiple sequences with an average sequence $\hat{T} = \arg\min_{\hat{T}} \{\text{DTW}(T_i, \hat{T})\}_{i=1}^{N}$ using the DTW distance. But it does not have a contribution to improving the alignment of DTW algorithm. Keogh and Pazzani [14] propose to replace the coordinator distances by the derivative distance (DDTW). Therefore, points with similar changing trend are matched. Jeong et al. [22] propose to weight the match of two points by the length of stretch such that the long shift is penalized (WDTW). Batista et al. [12] propose a similar penalty-based distance. That method additionally considers a coefficient on top of the DTW distance in order to avoid small distance between sequences of different complexity (CID-DTW). Like DTW, DDTW, WDTW and CID-DTW still have the limitation in considering the sequential nature of data. In [27,28], the DTW distance among sequences is employed as the features for further classification. Other studies focus on improving the efficiency of DTW [11] but do not improve the alignment results of DTW.

Our method DSW is similar to methods adjusting the pointwise distance. But DSW is different from previous methods in that DSW is able to utilize the sequential dynamic information of sequences. From Eq. (1), it is clear that given the distance matrix D, the basic subroutine of DTW is the pointwise Euclidean distance of temporal axis coordinate values. The point-to-point distance ignores local autocorrelation structure information. This is the key bottleneck of the classification performance of DTW. In this paper, we are going to offer a remedy by taking advantage of the memory ability of recurrent networks to learn a state representation for each time point. The alignment is performed in the state space of sequences instead of the time domain.

2.2 Reservoir computing models

Recurrent neural network (RNN) takes account of the past input history and the current input at each time point. This recursive nature makes it useful in preserving the autocorrelation structure of sequential data, such as language modelling [29], sequence analysis [5] and video processing [4].

Reservoir computing model [30] is a kind of RNNs having a fixed, randomly generated state transition modular, called the reservoir, and a trainable read-out layer. The reservoir provides dynamic features of input series, and the read-out is trained to generate desired outputs by assembling reservoir features. Popular reservoir computing models include Echo State Network (ESN) [31] and Liquid State Machine [32], etc.

In [3,6,7,33], the read-out weights of the ESN are proved to be able to extract discriminative features from the whole sequence. The time series distance is calculated as the distance of the read-out mappings [3,6,7]. Similarly, Fisher kernel [34] learns vectorial Fisher score as a representation for the whole sequence using a probabilistic generative model, e.g. HMM. Fisher kernel assumes that similar objects stretch the generative model parameters to a similar extent. In Fisher kernel, all the sequences have to be approximated by only one generative model, which seems restrictive.

This current work employs the ESN to capture the dynamics of sequences, which is different from previous studies [6,7]. Our goal is to uncover the state transition track by learning state-space representations for each time points, rather than a representation of the whole sequence [3,34,35].

DTW is commonly employed by comparing time points in the time domain in previous sequence classification tasks. The comparison of raw time points may make DTW sensitive to noise and weakly interpretable [10]. Numerous methods have been proposed to improve the effectiveness or efficiency of DTW. However, little prior work has contributed from the aspect of incorporating dynamic features to enhance DTW for sequence classification.

As a remedy, in this work, we propose to learn temporal point representations for sequences. Reservoir computing models are useful for handling sequential data because of their implicit memory for learning compact hidden representations. The hidden representations preserved by the reservoir could be valuable in providing DTW with dynamic information of sequences. We learn flexible and versatile representations for each time point by using the reservoir network. The integrated information in the time point representations provides discriminative features. The proposed method makes use of the history information of sequences to make the alignment semantically more sensible and provides better classification accuracy in comparison with DTW.

3 Dynamic state warping

3.1 Echo state network

Sequences are structured objects that have correlations among different time points. DTW finds an optimal alignment between two sequences so that the cumulative distance is minimized. However, it has limitations in making use of the dynamic information.

We propose an algorithm DSW as a mitigation. In particular, the mechanism of DSW is a two-stage process. First, to effectively incorporate the dynamic information into DTW, we propose to employ a nonlinear yet efficient dynamic system, i.e. Echo State Network (ESN), as the general-purpose temporal filter to convert the time points into latent states in the ESN. In this way, we are able to obtain dynamic temporal features, which could be more sensible descriptors for alignment. In the second stage, the same alignment operation as that of DTW is performed to align the state trajectory sequences.

Echo State Network (ESN) is a kind of reservoir model. ESN is characterized by a non-trainable, high-dimensional and nonlinear dynamical reservoir and an efficiently trained linear read-out layer (usually by linear regression). The reservoir learns a representation for each time point by taking into account the previous input history and the current input. In a typical ESN, the reservoir is randomly generated under the constraint of the maximum eigenvalue being less than one. This is also called the echo state property. Loosely speaking, it requires that the initial inputs have little influence on the final state. It is usually implemented by first normalizing the reservoir weight matrix to have a unitary spectral radius. Then, we multiply it with a scaling parameter. The read-out layer assembles the state-space features of the reservoir to learn a function mapping from reservoir states to the output sequence. Linear regression is usually employed to learn the read-out function.

In this study, we are not going to count on the assembling ability of the read-out layer [6], but we will analyse its effect on the performance of DSW in the experiment (see Sect. 4.4.3). The approximation ability of the ESN reservoir helps leverage the autocorrelation of original sequences.

To justify the randomness of ESN, Rodan et al. [36] propose a topologically fixed reservoir, cycle reservoir with jumps (CRJ). CRJ reservoir is more constrained than a typical ESN reservoir. It connects the reservoir neurons in a unidirectional circle and allows fixed-length jumps on the circle. In this case, there are only three types of connections for CRJ reservoir, i.e. input, jump and cyclic connections. The network weights can be determined by three values $r = \{r_i, r_j, r_c\}$ for each kind of connection.

The motivations of choosing the reservoir network, especially the CRJ network, as the adaptive temporal filter include:

(1) Reservoir models can provide parsimonious state representations for sequence points efficiently ($O(N)$).
(2) Reservoir models are able to take advantage of the sequential dynamic information of sequences.
(3) Reservoir models require much less effort to tune the network compared with a typical RNN, which may require more data, training time, and could be trapped in local optima.
(4) The CRJ reservoir network has been shown to be competitive or better than conventional randomly constructed ESN [36].

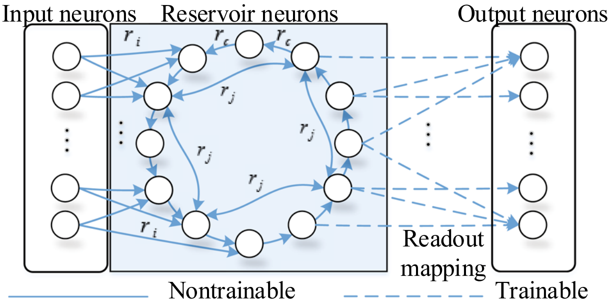


Fig. 5 Illustration for the CRJ reservoir network

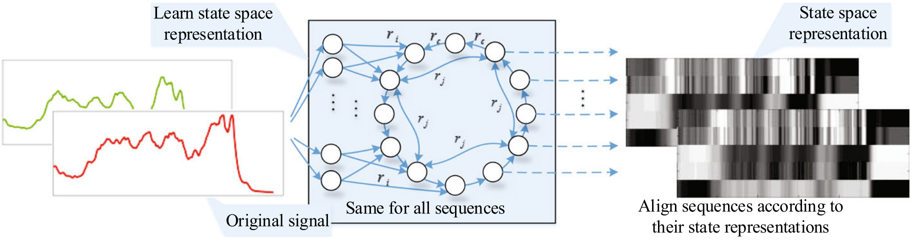

Learn state space representation
State space representation
r_i
r_c
r_c
r_j
r_j
r_j
r_j
r_i
Original signal
Same for all sequences
Align sequences according to their state representations

Fig. 6 Semantic demonstration of the key idea of DSW. The CRJ reservoir network converts the original sequences into state trajectory sequences. Then, two points are paired if their states are similar

3.2 The DSW algorithm

This paper takes CRJ as the base model for ease of analysis. Figure 5 illustrates the architecture of CRJ network. The mechanism of CRJ is generalized as:

$$\begin{cases} S(t+1) = g(\mathbf{R}S(t) + \mathbf{V}X(t+1)) \\ f(t) = g_{\text{out}}(\mathbf{W}S(t) + b) \end{cases} \tag{2}$$

where $S(t) \in R^N$ is the reservoir state, N is the number of neurons in the reservoir; $X(t) \in R^n$ is the input sequence, n is the number of input neurons; $\mathbf{R} \in R^{N\times N}$ is the reservoir weight matrix, $\mathbf{V} \in R^{N\times n}$ is the input weight matrix, $\mathbf{W} \in R^{O\times N}$ is the output weight matrix, $b \in R^O$ is the bias term, O is the number of output neurons; g is the state transition function, which is a nonlinear function and usually taken as tan$h$ or the sigmoid function; g_{out} is the activation function of outputs. Without loss of generality, in this paper, we fix $g = \tanh$ and g_{out} as the identity function.

The DSW algorithm is demonstrated in Algorithm 1. Figure 6 gives an illustration of the key idea of this paper.

Given an input sequence $X \in R^{L_X\times d}$, we first drive it through the nonlinear dynamic reservoir and obtain a state transition trajectory sequence $S \in [-1, 1]^{L_X\times N}$, where N is the reservoir size, according to Eq. (2). It transforms from the previous state into a new state in the state space given a new input time point. In this way, the temporal signal space sequence is converted into multivariate state-space sequences. The state trajectory sequence encapsulates the generating mechanism of the original sequence by taking into consideration the previous state and current input. In particular, the reservoir state representation is the activation of reservoir neurons with different driving input sequences. The state-space repre-

Algorithm 1 Dynamic state warping

```
Input: Two sequences $Q \in R^{L_Q \times d}$, $C \in R^{L_C \times d}$; #neurons of the reservoir; jump length of reservoir; initial parameters for network ($\{r_i, r_c, r_j\}$) (Table 1).
Output: A distance between the two sequences.
Use the CRJ network to convert the original sequences into reservoir state space according to Eq. 2.
Compute state point-level distance matrix $D \in R^{L_Q \times L_C}$.
Search the optimal warped path $\alpha$ and $\beta$ using dynamic programming.
Return the accumulative distance (Eq. (1)) as the distance between two input sequences.
```

sentations provide discriminative features with more versatility and flexibility than original sequences.

We then employ DTW to align the state trajectory sequences as usual. It prefers to align points on two sequences with similar states.

Therefore, the difference between DTW and DSW lies in that DTW uses distance between time points as a subroutine, while DSW uses the distance of the state-space representations.

Since our concern in this paper is to learn point-level states as representations, our framework can be naturally generalized to time-variant and time-invariant, univariate and multivariate series and employ different temporal filters.

Proposition 1 *The state sequence is able to scale the noise in the original sequence by a constant scaling.*

Proof The state sequence is able to reduce the noise in the original signal by constraining r_i^2. Given an additive noise ε in the sequence X, the distance between the state sequences is then:

$$\begin{aligned}
&||S^{\varepsilon}(t+1) - S(t+1)||^2 \\
&\quad = ||g(\mathbf{R}S(t) + \mathbf{V}(X(t) + \varepsilon)) - g(\mathbf{R}S(t) + \mathbf{V}X(t))||^2 \\
&\quad \leq ||\mathbf{R}S(t) + \mathbf{V}(X(t) + \varepsilon) - \mathbf{R}S(t) - \mathbf{V}X(t)||^2 \\
&\quad = ||\mathbf{V}\varepsilon||^2 \leq r_i^2 ||\varepsilon||^2,
\end{aligned}$$

where S^{ε} is the noisy state sequence. The above equation reveals that the noise in the state sequence is scaled by the input weight r_i^2. The nonlinear state transition function g can be tan$h$ or sigmoid function. Note that their derivative is not larger than one; thus, we have tan$h(\delta) \leq \delta$. That is, the noise level is reduced in the state sequence by adjusting r_i (usually $r_i \leq 1$ in the typical ESN setting [33]). □

In order to exploit the dynamic information in sequences, one might partition the original sequences into short segments and employ the distance of segments as a robust subroutine in DTW. This can also be viewed as a “kind” of smoothing filter. However, the segmentation of time series requires additional efforts on parameter selection and remains unclear how to segment the original time series to maintain the long or short autocorrelation information. By employing the ESN network, DSW is able to exploit the dynamic information of sequences with a fading memory of the past input history. ESN inherits the benefits of recurrent neural network keeping the appreciate dynamics, and it maintains a low computation complexity, which is the reason to use ESN instead of traditional recurrent neural networks.

To integrate the dynamic information, one can of course resort to other noise filters [37], such as Kalman filter [38] or low-pass/high-pass frequency filter [39], according to the sta-

Table 1 The parameter settings for DSW in our experiments

$\{r_i, r_c, r_j\}$	J	N	Spectral radius	Input dim
[0.01, 1]	$[2, \frac{N}{2}]$	5	0.85	2

The parameters are set following uniform distribution [36]. The spectral radius is the maximum eigenvalue of the reservoir weight matrix [31,40] (Sect. 4.4.1). The input Dim refers to the number of consecutive time points that are fed into the network (Sect. 4.4.5)

tistical characteristics, frequency, or distribution of sequences, if one deems them useful. Actually, these filters are also dynamic systems and comply with our framework. However, such knowledge about the error distribution or other characteristics in the sequence is often not available or varies among different subjects. For example, Kalman filter is only applicable to linear dynamic signals with Gaussian noise, which limits its application in many real-world problems. Similarly, for the frequency-based filter, the mechanism of incorporating the dynamic information is un-intuitive and may not generalize to time series datasets with different characteristics.

The key idea of this paper is to represent each time point by employing a nonlinear dynamical system with fading memory as the general-purpose nonlinear temporal filter. Providing the fixed general temporal filter is determined properly, each time point could be represented by encoding the generating mechanism and short-term/long-term autocorrelation information of the time series, to reflect the dynamic behaviours. The employed general-purpose nonlinear adaptive filter could be suitable for a variety of time series datasets.

4 Experiment

In this section, we perform extensive experiments to evaluate our method. In particular, this section is divided into three parts.

1. The robustness is evaluated on synthetic noisy data. The scalability to long series is tested by consistently increasing the length of sequences (Sect. 4.2).
2. We evaluate DSW by the classification results on 85 UCR time series datasets [21] in comparison with ED, DTW and other algorithms (Sect. 4.3).
3. We analyse the related properties of our method to provide more insights into DSW and its parameter sensitivity (Sect. 4.4).

4.1 Experimental set-up

We compare DSW with Euclidean distance and DTW. Since our concern here is the effectiveness of states as representations for time points, we do not refine alignment constraints for dynamic programming process. However, since the second stage of our method is intrinsically DTW, more advanced strengthening techniques for DTW, such as those proposed in [11,22], can be easily incorporated. The sequences have been normalized to have zero mean and unit standard deviation [11]. Without explicit mention, the reservoir of DSW is randomly generated. The related parameters are set following Table 1. The generalization error [14,21,27,37], i.e. the proportion of wrongly classified test samples, is employed for evaluation.

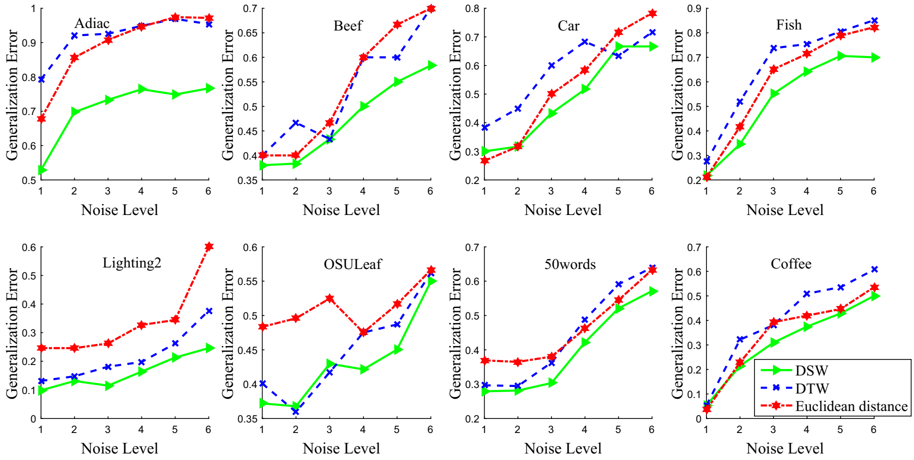


Fig. 7 The Generalization performance of Euclidean, DTW and DSW distance with 1NN classifier when facing noisy data. The result demonstrates that DSW is more robust than Euclidean distance and DTW

4.2 Robustness and scalability

4.2.1 Robustness

To evaluate the robustness of different distance measurements, we conduct experiments varying the noise in the dataset. In detail, we choose eight benchmark datasets [21], as indicated in Fig. 7. We add a Gaussian white noise into these datasets with zero mean and standard deviation varying in the set { 0.1,0.3,0.5,0.7,0.9,1.1}. The standard deviation is set in this manner in order to clearly present the tendency. On each noise level, we perform ten runs and report the average. The generalization error rate is computed using 1NN classifier with the ED, DTW and DSW.

The results are presented in Fig. 7. It shows that DSW is more robust to noise than compared methods. DSW consistently achieves smaller generalization error rates than DTW on most datasets. DTW is sensitive to noise because its basic unit is to compute the pointwise distance between time points and counts heavily on the sequence shape during alignment. Yet, the Gaussian noise changes the magnitude of original sequences. This partially explains the noise sensitivity of DTW. Euclidean distance maintains intermediate performance. When the noise level is low, Euclidean distance sometimes maintains slightly better performance than DTW and DSW. We presume the small quantity of additive zero-mean Gaussian noise may have less influence on the lockstep distance than warping based distance if the time series is roughly aligned. For DTW and DSW, the alignment may be influenced by the slight disturbance in this case.

4.2.2 Scalability to long sequences

In this section, we employ synthetic datasets containing varying length sequences to evaluate the scalability of DSW.

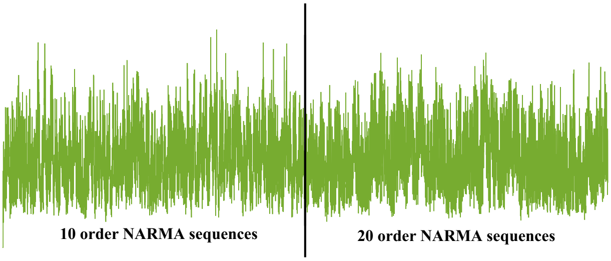


Fig. 8 Illustration of 10 order and 20 order NARMA synthetic sequences

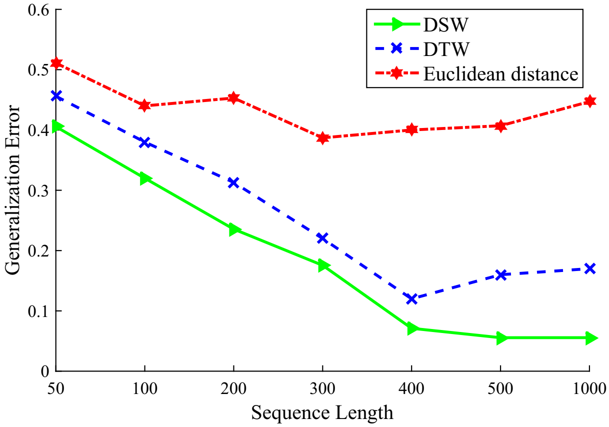


Fig. 9 The generalization trend of three distance measurements facing time series of different length

In particular, we generate a series of 10 order and 20 order NARMA sequences:

$$s(t+1) = 0.3s(t) + 0.05s(t)\sum_{i=0}^{9} s(t-i) + 1.5u(t-9)u(t) + 0.1$$

$$s(t+1) = \tan h(0.3y(t) + 0.05y(t)\sum_{i=0}^{19} y(t-i) + 1.5u(t-19)u(t) + 0.01) + 0.2,$$

where $s(t)$ is the output sequence and $u(t)$ is the input sequence and is generated in the range [0,0.5] with uniform i.i.d. distribution. These two kinds of sequences are generated using the same input sequence. The 10 order and 20 order NARMA sequences are treated as two classes. The original sequences are illustrated in Fig. 8.

We use the NARMA model of 10 order and 20 order, respectively, to generate synthetic data with varying sequence length. This experiment aims to present what will happen when the sequence length is becoming longer and longer. In detail, we generate 10 order and 20 order NARMA sequences of the length in {5000, 10000, 20000, 30000, 40000, 50000, 100000}. For each length, a generated sequence is then divided into non-overlapping subsequences of equal length to obtain 50 subsequences. We randomly select 25 sequences of 10 order and 25 sequences of 20 order as the training set of size 50. The rest 50 subsequences are used as test set. We use the 1NN classifier to classify the test set using the training set. For each length, the synthetic dataset is generated ten times and the classifica-

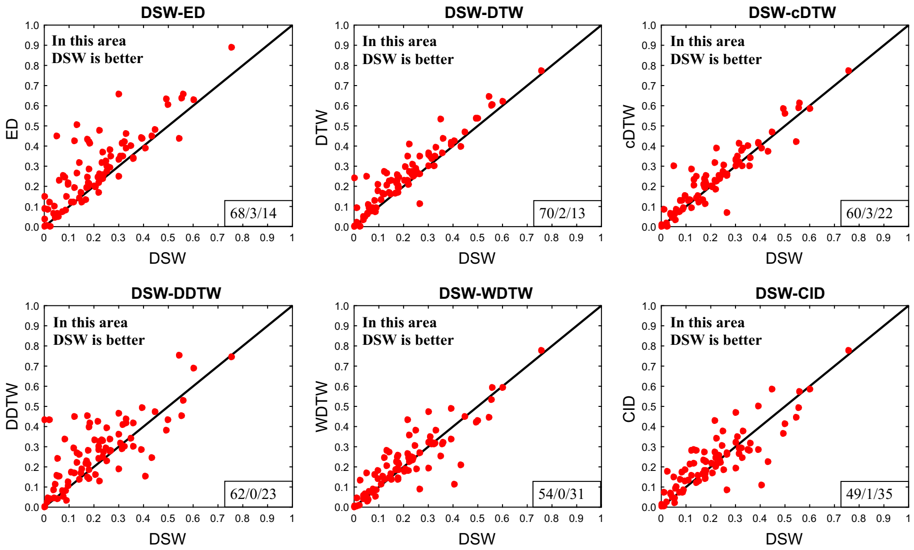


Fig. 10 The pairwise comparison of 1NN classification performance of DSW, ED, DTW, cDTW, DDTW, WDTW and CID-DTW on 85 UCR time series datasets. The numerical value at the bottom right means the number of wins/ties/losses of DSW in comparison with a compared method on 85 datasets. It is observed that DSW is better than compared methods on most datasets

tion is also repeated ten times. The average generalization error rate is collected as the final result.

Figure 9 demonstrates the performance of different distance measurements in terms of varying sequence length. From Fig. 9, we can make three main observations: (1) it shows that Euclidean distance scales poorly to long sequences. When the sequence length exceeds 400, Euclidean distance begins to degrade the generalization performance of the 1NN classifier. This result is as expected and consistent with our intuition that Euclidean distance is not suitable for long series. (2) The performance of DTW initially improves when facing long sequences. Its performance is much better than Euclidean distance. This result explains why DTW is so successful in sequence processing domains [9,11]. Sequences are usually long, and the time points are usually much larger than the number of observations. However, when sequence length grows longer than 600 points, the performance of DTW begins to worsen. (3) Unlike the compared distance measurements, DSW is consistently improving the performance in our experiment.

4.3 Classification performance on benchmark datasets

Datasets Our experiments are performed using the UCR time series datasets [21]. The 85 UCR datasets [21] are collected from different domains such as insect recognition, medicine, engineering, motion tracking, image and synthetic data. The datasets have already been divided into training and test set. The length of sequences varies from one dataset to another, with the minimum length 24 and maximum length 2709. In each dataset, the sequences are of equal length. The size of datasets varies from 24 to 8926. The number of classes varies from 2 to 60. Detailed information about the datasets is available on the website [21].

Experiment set-up and parameter settings Our results are averaged over ten repetitions. In each repetition, the network is optimized by selecting one network from 20 randomly initialized networks using leave-one-out cross-validation (LOOCV) on the training set. The parameter settings in Table 1 are employed. This set-up is applied to all datasets (as recommended in Table 1).

In our experiment, the network is selected in this arbitrary manner. Thus, the results may not be optimal for our method. We can of course design specific networks for different datasets by using the result of Sect. 4.4. We do not optimize the classification result by using more advanced strategy to simplify operations. However, when dealing with real-world problems, one can of course choose a better parameter setting using simple search strategies, e.g. grid search. For example, on BeetleFly dataset, we search the eigenspectra scaling using the range reported in Sect. 4.4.1 and improve the accuracy of DSW by 15%. Indeed, our results show that even using the randomly generated reservoir and our LOOCV strategy can yield surprisingly good results.

The Euclidean distance and DTW algorithm are determined; thus, only one run is performed on each dataset.

Compared methods The seven compared methods are:

(1) **ED**: the distance is calculated by Euclidean distance. The label of the nearest neighbour of the query is returned as the prediction label.

(2) **DTW**: The distance is computed using DTW. The predicted label is taken as the label of the nearest sequence in the training set.

(3) **cDTW**: cDTW learns the best warping window width on the training data for DTW. The optimal window size for each UCR dataset can be found on the website http://www.cs.ucr.edu/~eamonn/time_series_data/.

(4) **DDTW** [14]: The first order distance is calculated for DTW as in [14]. Points with similar derivative are more likely to be matched. DDTW can be viewed as a kind of smoothing for original time points by employing the derivative. This motivates us to compare DSW with DDTW, because DSW can be viewed as smoothing the original time points with the states of a nonlinear dynamic system.

(5) **WDTW** [22]: A multiplicative weight penalty is incorporated to reduce the warping degree. The distance between two points on two series is then $w_{|i-j|}(Q^i - C^J)^2$. Following [22], we employ a logistic function for the weight, i.e. $w_a = \frac{1}{1+g(a-L/2)}$, where $a = 1, 2, \ldots, L$, L is the sequence length and g is set on the training set.

(6) **CID-DTW** [12]: As indicated in [12], a complexity penalty is multiplied on the original DTW distance to take into account the difference in the complexity of two series. The warping window width of DTW distance employs the optimal value.

(7) **DSW**: The training sequences are converted into state sequences using a reservoir network. When a query sequence arrives, it is first converted into the state sequence using the same network. The distance between the query sequence and training sequences is computed by DTW using the state sequences. The network parameters, e.g. the connection weights, are initialized randomly following Table 1.

Platform The experimental environment is Matlab 2015a on an Intel Pentium Quad-Core G2120 3.10GHz CPU with 8GB RAM.

Using the parameter-free 1NN classifier enables us to compare the generalization error rates of DSW with compared algorithms in a pairwise fashion. The classification result on 85 UCR datasets is presented in Fig. 10.

It is clear that our method achieves lower classification error rates than compared methods. In detail, DSW achieves better performance on 68 out of 85 datasets than Euclidean distance.

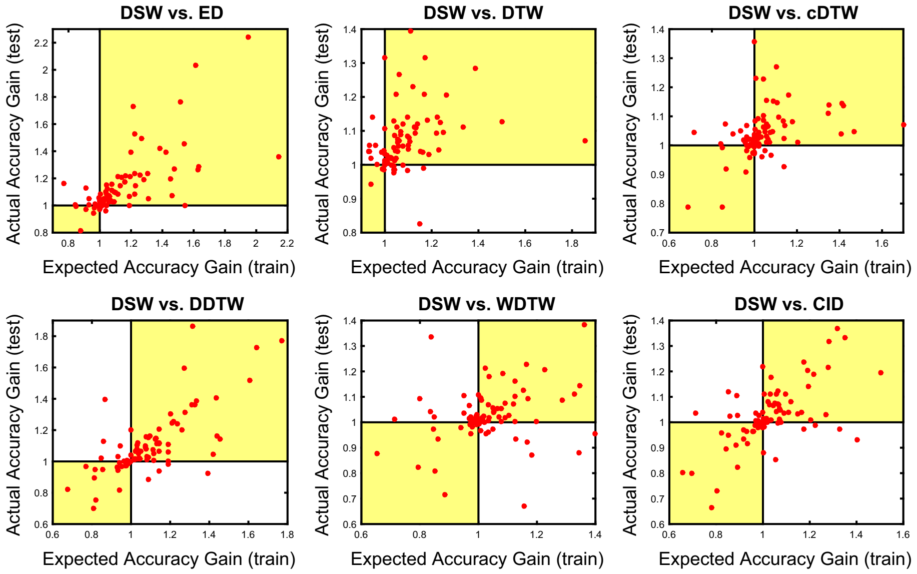


Fig. 11 Texas sharpshooter plot: expected performance gain on the training set versus actual performance gain on the test set. On most datasets, we can correctly predict the performance of DSW. TP: performance gain on both training and test set; TN: performance reduction in both training and test set; FN: performance reduction in training set but performance gain on test set; FP: performance gain on training set but performance reduction on test set. The FP region means that DSW is over-optimistic about its performance and is not desired. We only observe {4,7,6,4,12,10} datasets fall into the FP region for each compared methods

It has better performance on 70 datasets than DTW. The number of wins/ties/losses of DSW compared with other methods on all datasets is also reported in Fig. 10. On most datasets, DSW can yield good performance. Note that even though DSW does not have the warping window constraint of cDTW, the derivative information of DDTW, the weight decay of WDTW or the complexity term of cDTW, DSW still achieves satisfactory performance. One can of course add the warping window, derivative, weight decay or the complexity term into the formation of DSW for better performance. Our purpose here is to demonstrate that the states of the reservoir could provide versatile representations for original time points.

Texas sharpshooter plot We have evaluated the performance of DSW compared with Euclidean distance and DTW. The result is encouraging. Note that it would make no sense if we cannot know ahead of time whether DSW will perform well on a given dataset [12]. For this purpose, we employ the Texas sharpshooter plot to visualize whether DSW is useful by predicting the generalization ability using the classification results on the training set.

Let us take algorithm A and algorithm B as an example. In detail, we use the LOOCV performance on the training set of algorithms A and B to compute the expected performance gain: training accuracy of A/training accuracy of B; we use the accuracy on the test set to compute the actual performance gain: test accuracy of A/test accuracy of B. The result for all compared methods and DSW is presented in Fig. 11. The false positive (FP) region represents datasets on which we predict a performance gain but actually observe a performance reduction. The FP region is not desired. From Fig. 11, we observe that 4.7% points are in the FP region of DSW versus ED; 8.2% points are in the region of FP for DSW versus DTW. This result indicates that we have a very low probability to be over-optimistic on our prediction.

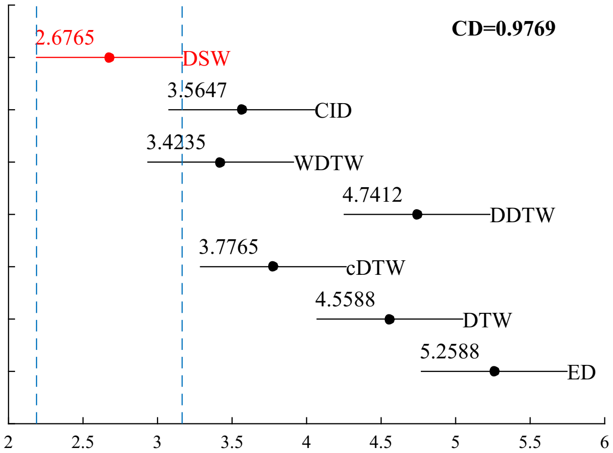


Fig. 12 Critical difference diagram for compared methods. The numerical values indicate the average rank on 85 UCR datasets. The difference is significant for algorithms with non-overlapped CDs. DSW performs significantly better than ED, DTW, cDTW and DDTW according to the Nemenyi test under the significance level 0.05. It is not surprising to find DSW is not significantly different from CID-DTW, as CID-DTW has employed optimal warping window and complexity information. The WDTW also fine-tunes the alignment on the training data

Statistical analysis To provide more insights into the performance of DSW compared with ED and DTW, etc., we perform statistical significance test for the difference of all approaches. In detail, the generalization error rates of the compared methods on all 85 UCR datasets are first converted into ranks. The algorithm with the minimum generalization error obtains the rank of 1, and the second minimum gets the rank of 2, etc. The lower rank indicates better performance. Then, we average the ranks across all datasets. We employ the Friedman test to compare the different distance measurements. The Friedman test indicates that the three distance measurements indeed behave differently. A post hoc pairwise Nemenyi test is performed to evaluate the significance of the rank differences. The result is demonstrated in Fig. 12. The main observation is that DSW gets the lowest rank and is the best distance measurement of the compared methods for classification. We also observe that our method is significantly better than most compared methods.

Up till now, we have compared DSW with Euclidean distance and DTW-related methods to demonstrate the effectiveness of our method for sequence classification. Next, we will empirically analyse the influence of the relating properties of reservoir model on DSW.

4.4 What is happening in the reservoir?

The sceptical readers may be wondering how a randomly generated reservoir network could achieve excellent classification performance. In this subsection, we are going to empirically uncover some insights about the reservoir network of DSW. In particular, we will analyse the spectral radius scaling of reservoir (Sect. 4.4.1), the input connection weights (Sect. 4.4.2), the predictability of reservoir for input sequences (Sect. 4.4.3), the size of reservoir (Sect. 4.4.4) and the input dimensionality (Sect. 4.4.5). Our strategy to evaluate these properties is to fix the other parameters and only vary the target property. The generalization error of DSW is calculated by the 1NN classifier.

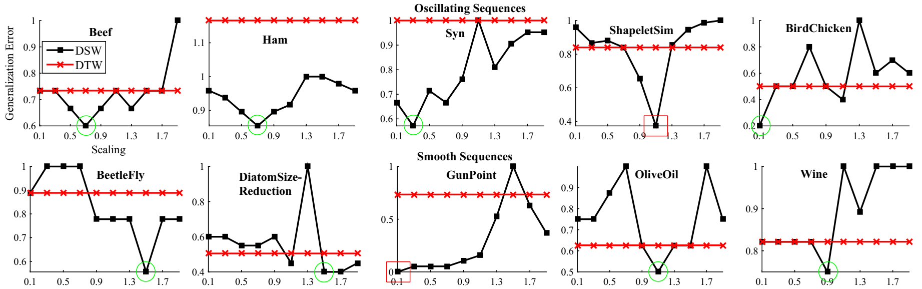


Fig. 13 Generalization error with varying reservoir scalings. The original error rates on each dataset have been divided by the maximal error rate for better visualization. The top row shows the result of fast oscillating sequences, and the bottom row shows the result of the smooth sequences. It demonstrates that (1) oscillating sequences usually need a small scaling and smooth sequences usually need a large scaling (see green circle); (2) local discriminative features need small scaling and global discriminative features need large scaling (see red rectangle). In addition, we plot the DTW generalization error for comparison. Note that DSW's parameters are set arbitrarily without optimizing by cross-validation. Despite the pessimistic set-up of DSW, on most datasets DSW maintains lower error than DTW (color figure online)

4.4.1 Spectral radius scaling

Previous studies have revealed that the spectral radius scaling should be less than one to guarantee echo state property [33]. That is, sequence modelling should be not sensitive to the initial value of the sequence. In particular, we have $\mathbf{R} = \text{scaling} \times \mathbf{R}/\max(\text{eig}(\mathbf{R}))$, where $\mathbf{R}$ is the reservoir weight matrix [33].

To study the effect of the scaling on the classification performance of DSW, we fix the reservoir parameters and then vary the spectral scaling parameter. In detail, let $r_i = 0.2$, $r_c = 0.5$, $r_j = 0.4$, $N = 5$ and jump length $= 2$. The scaling parameter varies from 0.1 to 1.9 with step size 0.2.

We artificially divide some UCR datasets into two groups according to whether the sequences are fast oscillating ones or slowly changing ones. Fast oscillating series changes rapidly over time, while the opposite is slowly changing series that evolves smoothly over time. The separation of the two groups is performed visually since it is difficult to give a definition to separate them. The aim of doing so will be clear later.

Generally, there is a trade-off between the input and the previous state in learning the current state representation. In particular, smaller scaling parameter weights the previous state less and gives more importance to the current input, resulting in short short-term memory [33]. On the opposite, large scaling endows more influence to the previous state, which leads to long short-term memory. Oscillating sequences usually need a short short-term memory to model the fast dynamics. Thus, small scaling parameter is preferred. Smooth sequences warrant a long short-term memory to take into account far earlier time points. It usually needs a larger scaling parameter.

Figure 13 presents the result of the generalization error rates varying with different scalings. The upside row is for oscillating series, and the downside row is for smooth series. It is clear from Fig. 13 that severely oscillating sequences usually require a small spectral scaling, while sequences with slow dynamics need a larger scaling. However, GunPoint dataset and ShapeletSim dataset show contrary result. For example, on GunPoint dataset, which contains smooth sequences, yet the classification result is optimal when the scaling is small. ShapeletSim is a dataset which contains oscillating sequences but warrants a large scaling.

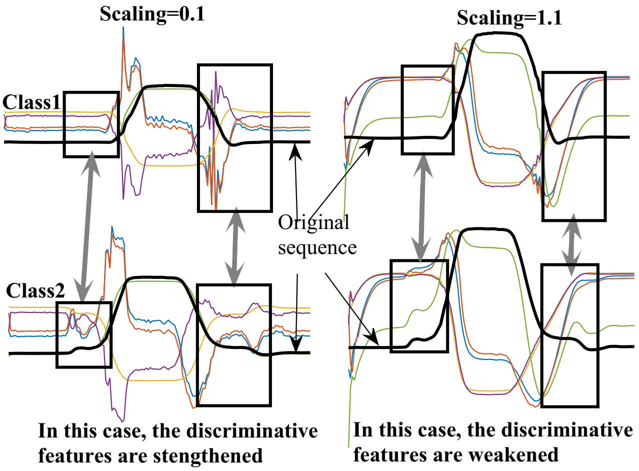


Fig. 14 State sequences (coloured) of original sequences (black) in GunPoint dataset. The scaling parameters are 0.1 (left) and 1.1 (right). The box shows the local discriminative features of two sequences (upside for class 1 and bottom for class 2). Small scaling is beneficial for finding local discriminative features. Large scaling is good for global discriminative features. Remind that the discriminative features for GunPoint dataset lie in a local area (see the box). The figure demonstrates that small scaling strengthens the discriminative features (left). Yet, large scaling makes the discriminative features weaken and thus hides them (right) (color figure online)

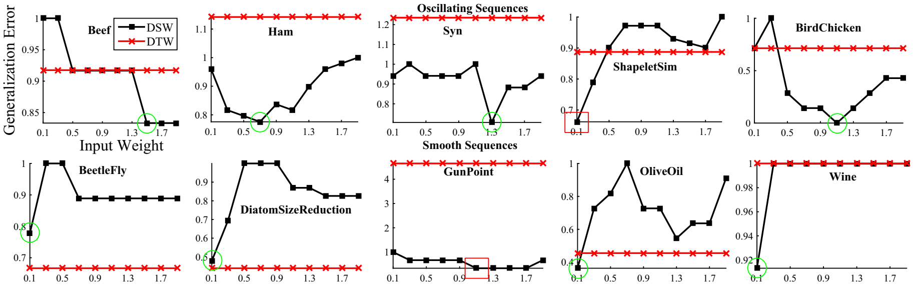


Fig. 15 Generalization error with varying input weights. The original error rates on each dataset have been divided by the maximum error rate for better visualization. The top row shows the result of fast oscillating sequences, and the bottom row shows the result of the smooth sequences. It demonstrates that (1) oscillating sequences usually need a large input weight and smooth sequences usually need a small input weight (see green circle); (2) local discriminative features need a small input weight and global discriminative features need a large input weight (see red rectangle). In addition, we plot the DTW generalization error as a comparison (color figure online)

We examine closely on GunPoint dataset. Figure 14 explains the observation. The GunPoint dataset contains two classes that differ from each other by a small region, i.e. taking a gun or no gun, as introduced in Sect. 1. The reservoir with a small scaling memory nearby points when converting time points into states. Therefore, the discriminative features are discovered. According to Fig. 14, for a small scaling parameter, the reservoir provides high dynamics enlarging the minor difference. On the other hand, when the scaling parameter is large, the reservoir representation is more smooth, hiding the small difference between two classes. To make the experiment reproducible, we have fixed the reservoir parameter as mentioned above.

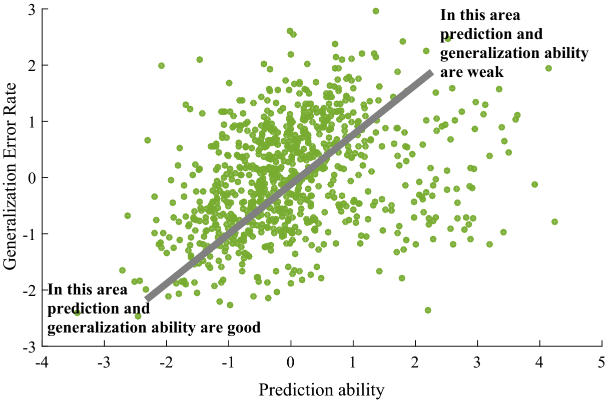


Fig. 16 The correlation between predictability of reservoir network and the generalization error rate of DSW. For visualization, the predictability and generalization error are normalized to have zero mean and unitary standard deviation

ShapeletSim dataset contains two classes of sequences, which oscillate severely. The two classes are generated with two frequencies, respectively, which is the main discriminative feature. The within-class sequences differ from each other by a phase shift, which results in their different shapes. To capture the inter-class difference, DSW needs long short-term memory to distinguish the two frequencies; thus, the result (Fig. 13) shows it performs best when the scaling is relatively large.

We find two main results in this experiment: (1) large scaling contributes to long short-term memory and small scaling contributes to short short-term memory. In DSW, this can be a guideline in designing reservoir for modelling sequences. (2) When the discriminative features concentrate on a small region of sequences, small scaling is preferred to strengthen them; when the discriminative features are global structures, large scaling is useful.

To summarize, for classification, the key concern is to discover the inter-class discriminative features. Normally, despite the specific warrants of some datasets, on most datasets, it is observed that fast oscillating dynamics benefit from small scaling parameter and more smooth sequences need larger scaling parameter.

4.4.2 Input weight

We fix the reservoir parameters as the same for the previous subsection. The input weight varies from 0.1 to 1.9 step by 0.2. Figure 15 demonstrates the results. It shows that small input weights are helpful for long short-term memory and large input weights for short short-term memory. This observation is consistent with the result of spectral scaling.

4.4.3 The predictability of reservoir model

We employ the predictability of read-out layer as a proxy for how well the state-space representation approximates the original sequences.

The reservoir model is trained to approximate the function mapping from input sequences to output sequences. Define the predictability of a reservoir as the difference between the empirical output and true output. To provide more insights into the reservoir model in DSW,

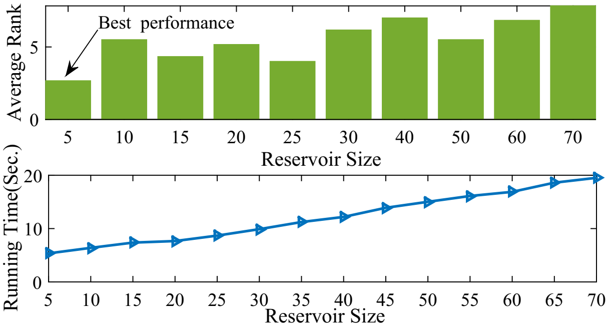


Fig. 17 The average rank on 42 old UCR time series datasets with respect to the reservoir size (upside). It demonstrates that small reservoir size is enough to yield good classification results for DSW. The bottom row presents the running time of DSW w.r.t. reservoir size

we analyse the relationship between the predictability on training set and classification performance on test set. We train the reservoir model using one-step forward prediction. Ridge regression is employed to learn the output weights [3,33]. The ridge regression parameter is selected in $\{10^{-5}, 10^{-4}, \ldots, 10\}$ by fivefold cross-validation [3]. In detail, we run ten repetitions for each of the 85 UCR datasets. The reservoir network is regenerated randomly every time. The predictability and generalization error of every network are recorded. We then normalize the two values to have zero mean and unit standard deviation.

By doing so, we can use the predictability of the network on training set as an indication of the classification performance on test set. We examine the relationship between the predictability on the training set and the classification error rate on test set. Figure 16 plots the correlation between these two values on all datasets. The Pearson correlation coefficient is 0.3342, which indicates these two variables are indeed correlated. Therefore, for DSW, it is important to obtain good classification performance by having a reservoir network that can approximate the original data well.

4.4.4 The size of dynamic reservoir

The number of neurons in the reservoir has an influence on the memory capacity of reservoir models [33]. Large reservoir provides more nonlinearity and dynamics. To study how the reservoir size affects DSW, we do experiments on 42 old UCR time series datasets with different reservoir sizes, ranging in$\{5, 10, 15, 20, 25, 30, 40, 50, 60, 70\}$. We fix the reservoir parameter as $r_i = 0.2$, $r_c = 0.3$, $r_j = 0.4$, $N = 5$ and jump length $= 2$. On each dataset, we obtain a set of generalization error rates corresponding to different reservoir sizes. The error rates are then sorted so that each value associates a rank. We record the average rank of classification performance on each reservoir size.

Figure 17 presents the experimental result. The upside row is the average rank for different reservoir sizes. Surprisingly, it shows that the reservoir size has very limited influence on the performance of DSW. The performance is good when the reservoir size is relatively small. In particular, it performs very well when the reservoir size is 5. It is reasonable to observe this result, since our task is to discriminate the examples in different classes instead of modelling the nonlinear dynamics. The bottom row presents the computational time on Beef dataset for different reservoir size. We observe that with a larger reservoir, more computational time is

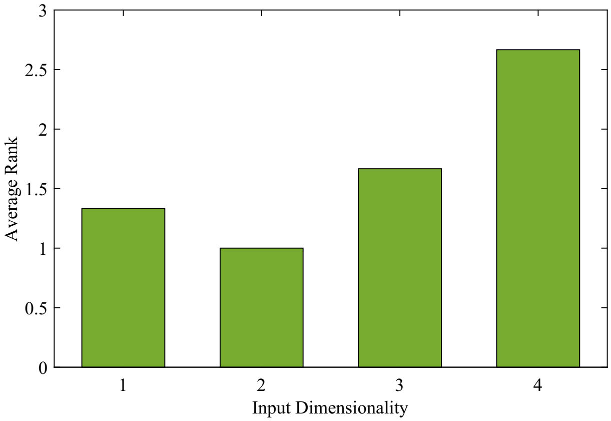


Fig. 18 The average rank of generalization error for different input dimensionality

needed. DSW is able to improve the classification performance with tolerable computational cost.

Note that in previous literature [33,36], the size of the reservoir is usually set as hundreds of neurons to capture the generating mechanism. In our study, we concern different aspects of sequences. The reservoir in our work is mainly to provide versatile dynamic features to help classification. As a result, we do not need too large reservoir size.

4.4.5 Input dimensionality

The dimensionality of the input influences the resulting state sequences. For n input dimensionality, it means we feed n successive time points as the input into the reservoir to obtain an updated state. We extend the original sequence by repeating the end of sequences n times so that the state sequences are of the same length as the original sequences.

We perform experiments on 42 old UCR datasets using different input dimensionality. On each dataset, the input dimensionality is selected as 1, 2, 3 and 4. Then using the selected input dimensionality, we learn reservoir state sequences. We have fixed the reservoir parameters during the experiment to guarantee only the input dimensionality is varied. The parameters are the same as that of Sect. 4.4.1. Figure 18 illustrates the average rank of each dimensionality over all 42 datasets. It clearly demonstrates that input dimensionality 2 achieves the best performance, followed by dimensionality 1. However, for dimensionality 3 and 4, it performs poorer.

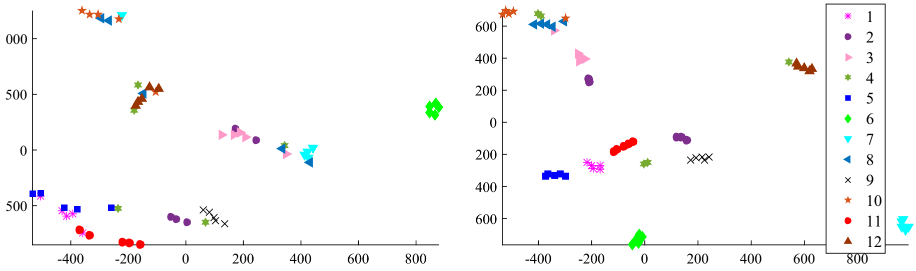
We summarize the above experiments.

(1) The classification performance of DSW is influenced by the trade-off between the input weights and the spectral radius and is proportional to the predictability of the read-out layer.
(2) The size of the reservoir network does not show significant influence, and a reservoir size of 5 often yields good performance.
(3) The proper selection of the input dimensionality would lead to better generalization.

In this paper, the default parameters have been used for all the experiments, whose performance can be further improved by optimizing proper parameters.

	DTW			PROCESS	DSW		
	1NN	P 1NN	P LR		1NN	P 1NN	P LR
Error rate	0.335	0.326	0.335	0.182	0.195	0.182	**0.169**

(a)



(b) **(c)**

Fig. 19 **a** Classification performance of DTW, PROCESS [27] and DSW on person identification dataset [27]. The warping window of DTW and DSW is not constrained. The reservoir network is determined by LOOCV on the training data. 1NN: classification with the distance (DTW or DSW) using 1NN. P 1NN and P LR: classification using the projected features with 1NN and LR classifier following [27]. The best result is boldfaced. **b**, **c** Visualizing DTW and DSW feature in 2D

5 Person identification by typing dynamics

We evaluate DSW on a real-world application that identifies the operator by the keystroke dynamics. The dataset [27] contains 548 multivariate time series, describing typing behaviours of 12 different users. Each user is asked to type the same text to collect the typing behaviour. The first 5 typing time series of each user are employed as the training data (60 in total), and the remaining series are treated as unseen test data (488 in total). We normalize the time series to have zero mean and standard deviation.

The reservoir network is randomly generated in DSW and selected by LOOCV on the training data. The classification result evaluated by the evaluation system (http://www.biointelligence.hu/typing-challenge/task2/) is reported in Fig. 19. The table presents the error rate of DTW, PROCESS [27] and DSW. Following [27], we employ the distance between a sequence and the training sequences as the feature for this sequence. Thus, each sequence is reformulated as 60-dimensional vectorial data (there are 60 sequences in the training set). DSW-induced features with logistic regression achieve the lowest error rate. The visualization of training data with this feature in 2D is performed by t-SNE [41]. In t-SNE, the initial dimension = 10, perplexity = 4. The figure demonstrates that the DSW feature is more separable than DTW feature.

6 Conclusion

In this paper, we propose a novel algorithm, DSW, for calculating the distance between sequences. DSW employs a reservoir network as a general-purpose nonlinear temporal filter. The original sequences are first converted into reservoir state trajectory sequences to capture the autocorrelation dynamic structure information. The state trajectory sequence provides versatile dynamical features for classification. Then, dynamic programming is performed to find an alignment between two state sequences. Therefore, points with similar states are matched.

We have conducted extensive experiments to evaluate DSW using both synthetic datasets and benchmark datasets, compared with DTW and its variants. The experimental results demonstrate that DSW achieves significant performance improvement with 1NN classifier. DSW is also empirically demonstrated to be endowed with better robustness and scalability. The running time of DSW includes the network training procedure by cross-validation and the alignment procedure. Similar to the constraint tuning of DTW, with the CV process for determining the network, DSW needs extra training time. The time complexity of alignment in DSW is predominantly $O(L_Q L_C)$, but can be reduced to linear time with constraints [11]. Our point here is to provide a novel strategy to consider dynamics as time states for calculating time series distance, rather than original time-domain points. For this purpose, we employ the dynamic states only and do not refine the align constraints, which could endow DSW with better classification performance and more efficiency.

We attribute the performance improvement of DSW to the smoothing ability of the dynamic temporal filter that the input history is implicitly stored in the hidden states of the dynamical system. Possible extensions of this work include: (1) to provide more efficient optimization methods for the reservoir network in DSW and (2) to design accelerating techniques for searching alignments in DSW.

Acknowledgements We would like to thank the anonymous reviewers for their constructive comments and suggestions to improve the quality of the paper. This work is supported in part by the National Key Research and Development Program of China (Grant No. 2016YFB1000905), the National Natural Science Foundation of China (Grant Nos. 91546116 and 61673363) and USTC Young Innovation Grant (WK2150110001).

References

1. Batal I, Cooper GF, Fradkin D, Harrison J, Moerchen F, Hauskrecht M (2016) An efficient pattern mining approach for event detection in multivariate temporal data. Knowl Inf Syst 46(1):115–150
2. Jo Y, Loghmanpour N, Rosé CP (2015) Time series analysis of nursing notes for mortality prediction via a state transition topic model. In: Proceedings of the 24th ACM international on conference on information and knowledge management, ACM, pp 1171–1180
3. Chen H, Tino P, Rodan A, Yao X (2014) Learning in the model space for cognitive fault diagnosis. IEEE Trans Neural Netw Learn Syst 25(1):124–136
4. Goroshin R, Bruna J, Tompson J, Eigen D, LeCun Y (2015) Unsupervised learning of spatiotemporally coherent metrics. In: Proceedings of the IEEE international conference on computer vision, pp 4086–4093
5. Pei W, Tax DM, van der Maaten L (2016) Modeling time series similarity with siamese recurrent networks. arXiv preprint arXiv:1603.04713
6. Chen H, Tang F, Tino P, Yao X (2013) Model-based kernel for efficient time series analysis. In: Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining, ACM, pp 392–400
7. Chen H, Tang F, Tino P, Cohn AG, Yao X (2015) Model metric co-learning for time series classification. In: Proceedings of the twenty-fourth international joint conference on artificial intelligence, AAAI Press, pp 3387–3394
8. Aminikhanghahi S, Cook DJ (2017) A survey of methods for time series change point detection. Knowl Inf Syst 51(2):339–367
9. Bagnall A, Bostrom A, Large J, Lines J (2016) The great time series classification bake off: an experimental evaluation of recently proposed algorithms. Extended version. arXiv preprint arXiv:1602.01711
10. Ye L, Keogh E (2009) Time series shapelets: a new primitive for data mining. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining, ACM, pp 947–956
11. Rakthanmanon T, Campana B, Mueen A, Batista G, Westover B, Zhu Q, Zakaria J, Keogh E (2012) Searching and mining trillions of time series subsequences under dynamic time warping. In: Proceedings

of the 18th ACM SIGKDD international conference on knowledge discovery and data mining, ACM, pp 262–270

12. Batista GE, Keogh EJ, Tataw OM, de Souza VM (2014) CID: an efficient complexity-invariant distance for time series. Data Min Knowl Disc 28(3):634–669
13. Berndt DJ, Clifford J (1994) Using dynamic time warping to find patterns in time series. In: AAAI workshop on KDD, Seattle, WA, vol 10, pp 359–370
14. Keogh EJ, Pazzani MJ (2001) Derivative dynamic time warping. In: SDM, SIAM, vol 1, pp 5–7
15. Zhou F, De la Torre F (2016) Generalized canonical time warping. IEEE Trans Pattern Anal Mach Intell 38(2):279–294
16. Kogan JA, Margoliash D (1998) Automated recognition of bird song elements from continuous recordings using dynamic time warping and hidden markov models: A comparative study. J Acoust Soc Am 103(4):2185–2196
17. Ding H, Trajcevski G, Scheuermann P, Wang X, Keogh E (2008) Querying and mining of time series data: experimental comparison of representations and distance measures. Proc VLDB Endow 1(2):1542–1552
18. Begum N, Ulanova L, Wang J, Keogh E (2015) Accelerating dynamic time warping clustering with a novel admissible pruning strategy. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, ACM, pp 49–58
19. Shariat S, Pavlovic V (2016) Robust time-series retrieval using probabilistic adaptive segmental alignment. Knowl Inf Syst 49(1):91–119
20. Ye L, Keogh E (2011) Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. Data Min Knowl Disc 22(1–2):149–182
21. Chen Y, Keogh E, Hu B, Begum N, Bagnall A, Mueen A, Batista G (2015) The UCR time series classification archive. www.cs.ucr.edu/~eamonn/time_series_data/
22. Jeong Y-S, Jeong MK, Omitaomu OA (2011) Weighted dynamic time warping for time series classification. Pattern Recogn 44(9):2231–2240
23. Faloutsos C, Ranganathan M, Manolopoulos Y (1994) Fast subsequence matching in time-series databases. In: ACM SIGMOD Record, ACM, pp 419–429
24. Lemire D (2009) Faster retrieval with a two-pass dynamic-time-warping lower bound. Pattern Recogn 42(9):2169–2180
25. Garreau D, Lajugie R, Arlot S, Bach F (2014) Metric learning for temporal sequence alignment. In: Advances in neural information processing systems, pp 1817–1825
26. Petitjean F, Forestier G, Webb GI, Nicholson AE, Chen Y, Keogh E (2014) Dynamic time warping averaging of time series allows faster and more accurate classification. In: 2014 IEEE international conference on data mining, IEEE, pp 470–479
27. Neubrandt D, Buza K (2017) Projection-based person identification. In: International conference on computer recognition systems, Springer, Berlin, pp 221–228
28. Meszlényi RJ, Hermann P, Buza K, Gál V, Vidnyánszky Z (2017) Resting state fMRI functional connectivity analysis using dynamic time warping. Front Neurosci 11:75
29. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp 3111–3119
30. Jaeger H (2002) Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the “echo state network” approach, vol 5. GMD-Forschungszentrum Informationstechnik, Bonn
31. Jaeger H (2001) Short term memory in echo state networks. Bonn, GMD-Forschungszentrum Informationstechnik
32. Natschläger T, Maass W, Markram H (2002) The “liquid computer”: a novel strategy for real-time computing on time series. In: Special issue on foundations of information processing of TELEMATIK, vol 8, no LNMC-ARTICLE-2002-005, pp 39–43
33. LukošEvičIus M, Jaeger H (2009) Reservoir computing approaches to recurrent neural network training. Comput Sci Rev 3(3):127–149
34. Maaten L (2011) Learning discriminative fisher kernels. In: Proceedings of the 28th international conference on machine learning, pp 217–224
35. Srivastava N, Mansimov E, Salakhudinov R (2015) Unsupervised learning of video representations using lstms. In: International conference on machine learning, pp 843–852
36. Rodan A, Tiňo P (2012) Simple deterministically constructed cycle reservoirs with regular jumps. Neural Comput 24(7):1822–1852
37. Höppner F (2017) Improving time series similarity measures by integrating preprocessing steps. Data Min Knowl Disc 31(3):851–878
38. Grewal MS (2011) Kalman filtering. In: Lovric M (ed) International encyclopedia of statistical science. Springer, Berlin, pp 705–708

39. Anissa S, Hassene S, Zouhair M (2013) Efficient speech denoising applied to colored noise based dynamic low-pass filter supervised by cascade neural networks. In: 2013 International conference on electrical engineering and software applications, IEEE, pp 1–5
40. Jaeger H (2001) The echo state approach to analysing and training recurrent neural networks-with an erratum note. Bonn, Germany: German National Research Center for Information Technology GMD Technical Report vol 148, p 34
41. Van der Maaten L, Hinton G (2008) Visualizing data using t-SNE. J Mach Learn Res 9(2579–2605):85

Zhichen Gong received a B.E. degree from Anhui University, Hefei, China, in 2015. He is currently a Master student in the UBRI laboratory at the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China. His research interests include machine learning, dynamical models and sequential data mining.

Huanhuan Chen received the B.Sc. degree from the University of Science and Technology of China (USTC), Hefei, China, in 2004, and the PhD degree in computer science from the University of Birmingham, Birmingham, UK, in 2008. He is currently a Full Professor at UBRI, School of Computer Science and Technology, USTC. His research interests include neural networks, Bayesian inference and evolutionary computation. Dr. Chen received the 2015 International Neural Network Society Young Investigator Award, the 2012 IEEE Computational Intelligence Society Outstanding Ph.D. Dissertation Award and the 2009 British Computer Society Distinguished Dissertations Award. He is an associate editor of the IEEE Transactions on Neural Networks and Learning Systems. He is a senior member of the IEEE.