Information Sciences 393 (2017) 91–107

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Dynamic Time Warping under limited warping path length

Zheng Zhang[a,*], Romain Tavenard[b], Adeline Bailly[b], Xiaotong Tang[c], Ping Tang[a], Thomas Corpetti[d,b]

[a] Institute of Remote Sensing and Digital Earth (RADI), Chinese Academy of Sciences (CAS), No.20 Datun Road, Chaoyang District, Beijing, 100101, China
[b] University of Rennes 2, UMR 6554 LETG-Rennes COSTEL, Rennes Cedex 35043, France
[c] Northeastern University, Qinhuangdao, Hebei, China
[d] CNRS, LETG-Rennes COSTEL UMR 6554 CNRS, Place du Recteur Henri Le Moal, Rennes Cedex 35043, France

ARTICLE INFO

Article history:
Received 11 February 2016
Revised 21 July 2016
Accepted 2 February 2017
Available online 9 February 2017

Keywords:
Dynamic Time Warping (DTW)
Time series
Distance measure
Warping path
Classification

ABSTRACT

Dynamic Time Warping (DTW) is probably the most popular distance measure for time series data, because it captures flexible similarities under time distortions. However, DTW has long been suffering from the pathological alignment problem, and most existing solutions, which essentially impose rigid constraints on the warping path, are likely to miss the correct alignments. A crucial observation on pathological alignment is that it always leads to an abnormally large number of links between two sequences. Based on this new observation, we propose a novel variant of DTW called LDTW, which limits the total number of links during the optimization process of DTW. LDTW not only oppresses the pathological alignment effectively, but also allows more flexibilities when measuring similarities. It is a softer constraint because we still let the optimization process of DTW decide how many links to allocate to each data point and where to put these links. In this paper, we introduce the motivation and algorithm of LDTW and we conduct a nearest neighbor classification experiment on UCR time series archive to show its performance.

1. Introduction

With the explosion of acquisition devices and sensors, time series is becoming more available in a very large number of applications (multi-media, physics, remote sensing, economics, clinical medicine, biodemography, etc) [21,29] and it has the advantage of fully capturing the changing process of the observed objects [14]. Therefore not only the information in a specific time t is important but also the temporal evolution of the observed processes is a key information to analyze. A fundamental task in time series analysis is to quantify the similarity (or dissimilarity) between two numerical sequences [11]. The performance of such a distance measure is critical to many data mining jobs, for instance, classification [17,20,28], clustering [26,29], retrieval [2,22,31], etc. In the context of time series, usual distance measures such as Euclidean distance, are often not suitable [11], because time shifts or time distortions exist commonly and unpredictably.

Dynamic Time Warping (DTW), introduced three decades ago in the context of sound processing [33], is a widely accepted distance measure for time series [11]. DTW is famous for its ability to manage time distortions by realigning time

* Corresponding author.
E-mail addresses: zhangzheng2035@163.com (Z. Zhang), romain.tavenard@univ-rennes2.fr (R. Tavenard), adeline.bailly@uhb.fr (A. Bailly), xiaotongtang1994@163.com (X. Tang), tangping@radi.ac.cn (P. Tang), thomas.corpetti@univ-rennes2.fr (T. Corpetti).

http://dx.doi.org/10.1016/j.ins.2017.02.018
0020-0255/

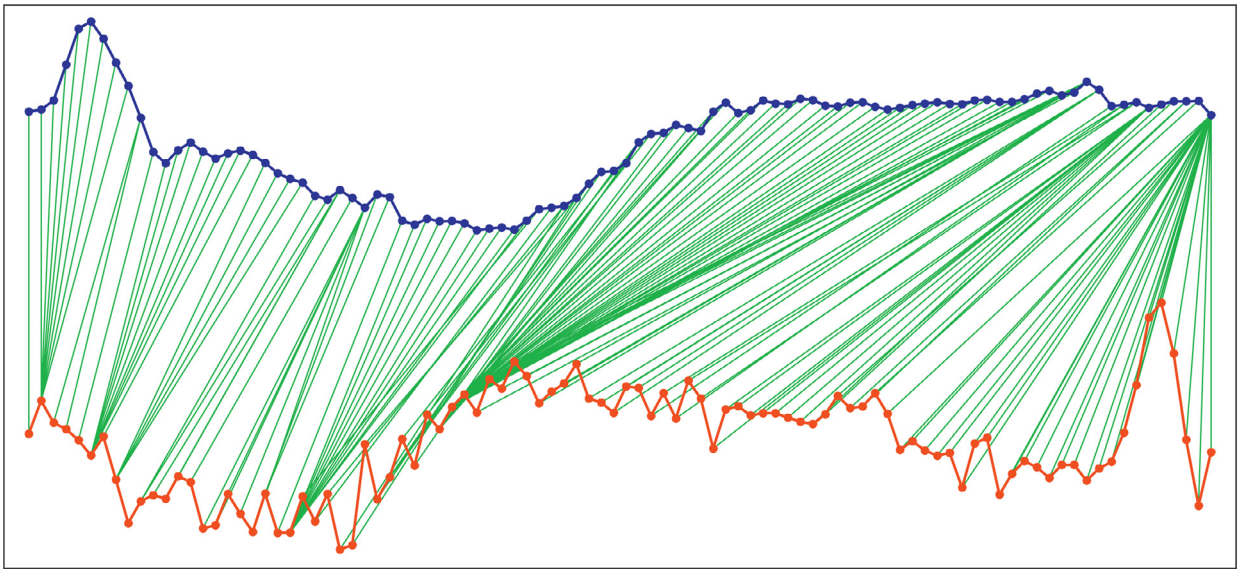


Fig. 1. A typical example of pathological alignment generated by DTW. The two time series here are the first two in the ECG200 training set.

series when comparing them. However, DTW still suffers from its drawback that it could lead to pathological alignments [23]. As an example, Fig. 1 illustrates a typical pathological alignment generated by DTW, where we can observe several singularities. A singularity is a data point inside one time series that links to a large subsection of the other time series. The presence of singularities is the main signature of pathological alignment. Obviously, such an alignment is not the "correct" one we could expect and by "correct" we mean intuitively obvious "feature to feature" alignment. The quality of this alignment greatly influences the accuracy of the associated similarity measure. We cannot find accurate similarity based on pathological alignment. To cope with this problem, an intuitive solution is to limit the number of points a point can link to, and this idea has been implemented by a widely used category of constraints on DTW, which is called windowing constraint [6,27,30,34] (will be discussed in Section 2.3). Although windowing constraint can solve pathological alignment, it also brings another problem that it takes a risk of preventing the correct alignment from being generated.

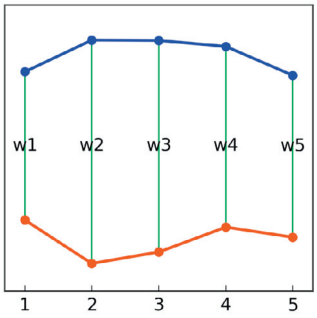
In fact, a crucial observation on pathological alignment is that if singularities happen, the total number of links (green lines in Fig. 1) between sequences must be abnormally large. Therefore, another intuitive solution, which will be explored in this paper, is to limit the total number of links between two time series. This way, we can also oppress singularities effectively. Moreover, this is a softer constraint than windowing constraint because we still let the optimization process of DTW decide how many links to allocate to each data point and where to put these links, instead of setting a rigid limit. As a result, it allows more flexibilities and avoid the risk of missing the correct alignment. We call this new method LDTW ("L" for "Limited" warping path "Length"), because it essentially limits the length of the warping path (detailed definition of warping path can be found in Section 2) during the DTW process.

LDTW originates from naive DTW. To clearly introduce LDTW and its motivations, we give a deep review of DTW and its variants in Section 2. Then in Section 3, we provide and explain the detailed implementation of the LDTW algorithm based on dynamic programming process. In the experiment section, we use LDTW as a distance measure and we report the one nearest neighbor classification error rate on UCR time series archive, in comparison with Euclidean distance, DTW without constraint and DTW with most popular windowing constraints. The experimental result shows that LDTW can achieve a better classification performance on most datasets. The upper bound of warping path length is the only parameter of LDTW and we also describe how to find the optimal upper bound by LOOCV (leave-one-out cross validation) [24] on the training set.

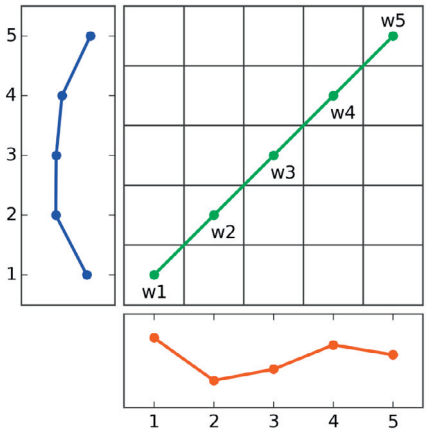
2. Dynamic Time Warping (DTW)

2.1. Notation

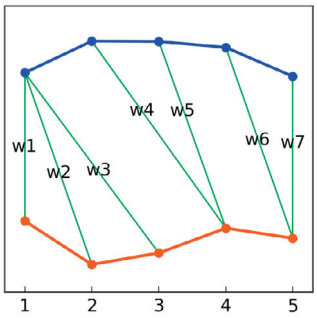
Before entering into technical details, Table 1 establishes a set of notations that will be used throughout this paper. Note that the relation between current step count s and current path length l is: $s = l - 1$, because the path length takes into account the starting point.



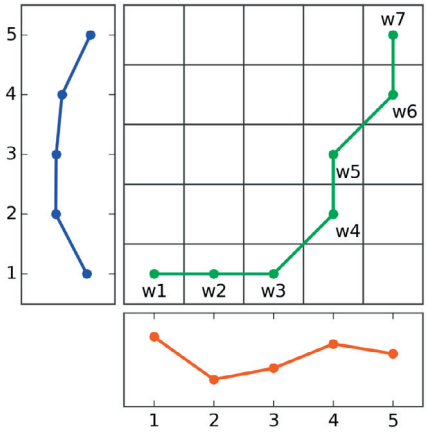
(a) Euclidean alignment



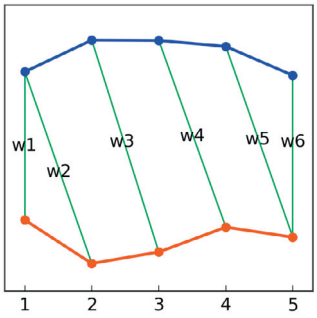
(b) Euclidean warping path



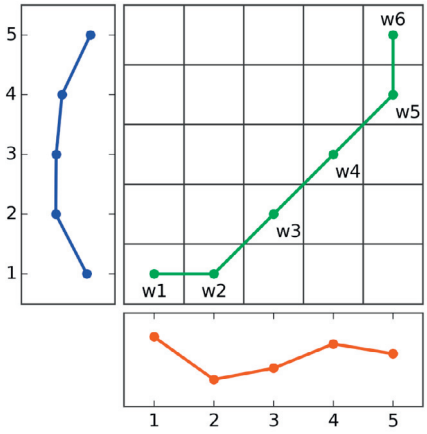
(c) DTW alignment



(d) DTW warping path



(e) LDTW alignment



(f) LDTW warping path

Fig. 2. Alignments and their corresponding warping paths for Euclidean distance, DTW, and LDTW.

Table 1
List of notations.

$A = a_1 \ \ldots \ a_r \ \ldots \ a_R$	- a time series of length R
$B = b_1 \ \ldots \ b_c \ \ldots \ b_C$	- another time series of length C
A_n	- a sub-sequence consists of the first n points of A
$m[r][c]$	- a $R \times C$ cumulative cost matrix used in DTW, each cell keeps only one value
$m[r][c][s]$	- a $R \times C$ cumulative cost matrix used in LDTW, each cell keeps multiple values
r	- row index of the cumulative cost matrix m, also the index of A
c	- column index of the cumulative cost matrix m, also the index of B
$W = w_1 \ \ldots \ w_l \ \ldots \ w_L$	- a warping path of length L
$w_l = (r, c)$	- a link between a_r and b_c
s	- current step count of the warping path, s starts from 0 and $s = l - 1$
L_{UB}	- upper bound of warping path length
$\delta()$	- cost between points of time series

2.2. Naive DTW

As shown in Figs. 1 and 2, DTW is alignment-based. The terminology alignment here refers to a collection of all links (or couplings) between two time series. The goal of DTW is to find an optimal alignment between two time series that achieves minimum global cost while ensuring time continuity. The global cost is the summation of the cost between each pair of points in the alignment. So we should first define the cost between points of time series, which is denoted by the symbol δ. Euclidean distance or squared Euclidean distance is normally used as the base cost δ and we employ squared Euclidean distance as our δ in this work. The detailed definition of δ is shown in Eq. (1).

$$\delta(a_r, b_c) = (a_r - b_c)^2 \tag{1}$$

The optimal global cost of DTW, namely, the DTW distance, can be recursively calculated by :

$$D(A_r, B_c) = \delta(a_r, b_c) + min\begin{cases} D(A_r, B_{c-1}) \\ D(A_{r-1}, B_c) \\ D(A_{r-1}, B_{c-1}) \end{cases} \tag{2}$$

A direct implementation of Eq. (2) leads to an exponential time complexity if we compute each subproblem from scratch. Fortunately, Eq. (2) belongs to a typical dynamic programming problem [10,12], because overlapping subproblems widely exhibit. For instance, the computation of $D(A_r, B_{c-1})$ and $D(A_{r-1}, B_c)$ both require $D(A_{r-1}, B_{c-1})$. The right solution here is to memorize the results of all these subproblems in a cumulative cost matrix, where the cell (r, c) keeps the result of $D(A_r, B_c)$. This way the process of DTW is converted into the process of fully filling the cumulative cost matrix in a certain order. Since the dimension of the matrix is $R \times C$, the time complexity of the DTW algorithm is $\Theta(R \times C)$. Algorithm 1 shows the

Algorithm 1. DTW.

Require: $A = \langle a_1 \ldots a_r \ldots a_R \rangle$
Require: $B = \langle b_1 \ldots b_c \ldots b_C \rangle$
Let $\delta()$ be the cost between two points of different time series
Let $m[r][c]$ be the cumulative cost matrix
function DTW(A, B)
  ▷ *Initializing the first column and row of m*
  $m[1][1] = \delta(a_1, b_1)$
  for $r = 2 to R$ **do**
    $m[r][1] = m[r-1][1] + \delta(a_r, b_1)$
  end for
  for $c = 2 to C$ **do**
    $m[1][c] = m[1][c-1] + \delta(a_1, b_c)$
  end for
  ▷ *Fully filling the cumulative cost matrix*
  for $r = 2 to R$ **do**
    for $c = 2 to C$ **do**
      $m[r, c] = \delta(a_r, b_c) + min\begin{cases} m[\ \ r\ \ ][c-1] \\ m[r-1][\ \ c\ \ ] \\ m[r-1][c-1] \end{cases}$
    end for
  end for
  return $m[R][C]$
end function

basic implementation of DTW. Intuitively, we fill the bottom row and the leftmost column first, and then sweep the matrix row by row (from left to right) from bottom to top according to Eq. (2). The last matrix cell $m[R][C]$ holds the final DTW distance between two entire sequences.

From the perspective of cumulative cost matrix, we can only solve the calculation of DTW distance. But if we want to gain a deep insight into DTW and the motivations of its variants, we need to review DTW closely from the perspective of warping path. As shown in Fig. 2, a warping path consists of a series of continuous matrix cells passing through the cumulative cost matrix. Each matrix cell, for example (r, c), corresponds to a link (or coupling) between two points A_r *and* B_c. In this way, the warping path W defines the alignment between two time series.

The warping path is generated by certain rules and different rules lead to different distance measures. From this point of view, the Euclidean distance can also be regarded as alignment-based and has its own rule of warping path. As shown in Fig. 2(a-b), Euclidean distance obeys a rigid one-to-one alignment and thus its warping path is only composed of the diagonal cells of the cumulative cost matrix. Implied by Eq. (2), the warping path of DTW, W, must satisfy the following three rules [23]:

1. Boundary constraint :
   $w_l = (1, 1)$ *and* $w_L = (R, C)$
2. Monotonicity constraint :
   Given $w_l = (r, c)$, $w_{l+1} = (r', c')$
   Then $r' \geq r$ *and* $c' \geq c$
3. Continuity constraint :
   Given $w_l = (r, c)$, $w_{l+1} = (r', c')$
   Then $r' \leq r + 1$ *and* $c' \leq c + 1$

The boundary constraint indicates that the warping path should start at the lower-left corner (1, 1) and end at the top-right corner (R, C). The other two constraints define the admissible step pattern, where only the upper, upper-right, and right neighbors of the current position are reachable at each step. Fig. 2(c-d) show the alignment and the corresponding warping path generated by DTW. If we follow the idea of LDTW and make the length of warping path not longer than six, then we get Fig. 2(e-f).

Given the warping path $W = \langle w_1 \ \ldots \ w_l \ \ldots \ w_L \rangle$, DTW distance can also be defined as :

$$D(A, B) = \min_{W} \left[\sum_{l=1}^{L} \delta(w_l) \right] \tag{3}$$

where if $w_l = (r, c)$, then $\delta(w_l) = \delta(a_r, b_c)$.

This new definition reminds us that the idea of DTW is to find the optimal warping path that leads to the minimum cumulative cost.

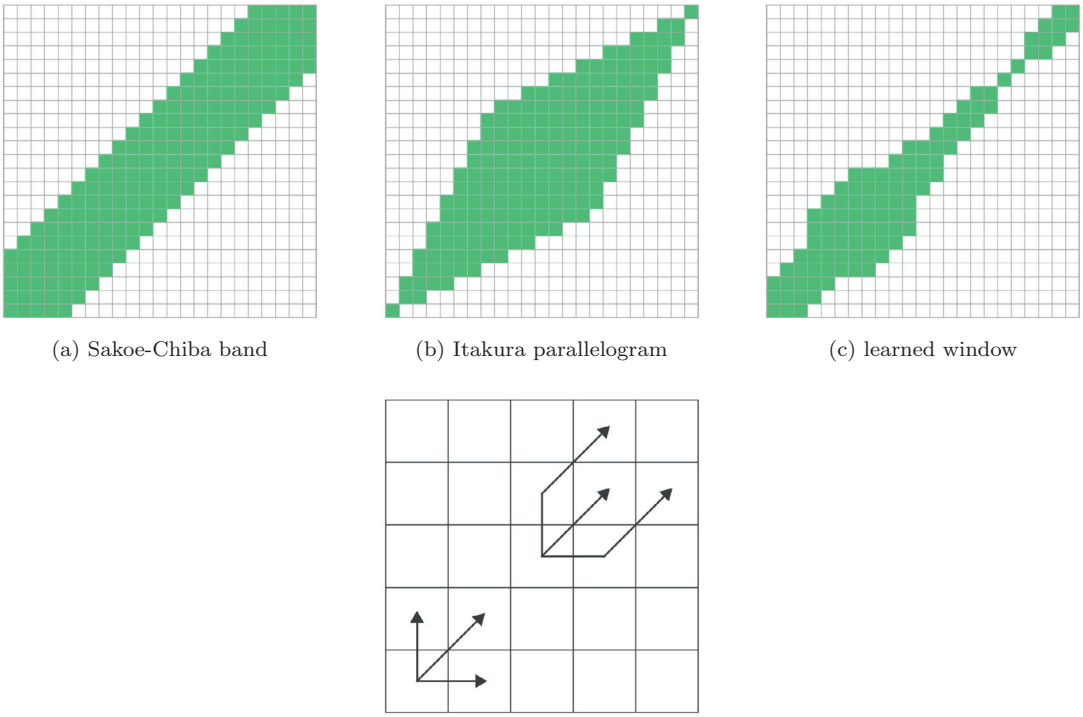
Unlike DTW distance, the corresponding warping path does not emerge explicitly during the DTW algorithm. Although the warping path has already been determined, it hides in the cumulative cost matrix and you need another backtracking procedure to reveal it. Starting from the upper-right corner (R, C), we check the three backward neighbors of the current position at each stage. Assume (r, c) is the current position inside the cumulative cost matrix. Its three backward neighbors, $(r-1, c)$, $(r, c-1)$, *and* $(r-1, c-1)$, will be checked and the one with the smallest value will be added to the warping path. Then we move to the new position and repeat the above steps until the lower-left corner (1, 1) is reached. Note that the warping path comes from backtracking so it is in reverse order. We need to reverse the entire path in the end to get the forward-going path.

2.3. Variants of DTW

As mentioned in the Introduction, DTW has the drawback of generating pathological alignments. Therefore, a large number of variants of DTW have been proposed to tackle with this issue. Generally, we can classify these variants into two major categories.

The first category essentially sets constraints on the warping path. The most widely used one is called windowing constraint [6,27,30,34]. From the perspective of alignment, it limits the number of points any point can link to, and from the perspective of warping path, it limits the allowable domain of the warping path in a specific area. Fig. 3(a-b) illustrate two most widely used shapes of constraint windows, Sakoe–Chiba band and Itakura parallelogram. In order to introduce more flexibility than a fixed authorized shape, some researchers prefer to learn an adaptive window shape from the data [7,32,35]. For instance, Fig. 3(c) shows a window generated by learning. Other types of constraints include step pattern [19,27] and slope weighting [25,33]. Step pattern specifies the reachable neighbors for each step, as illustrated in Fig. 3(d). Slope weighting sets a bias toward a certain direction.

The second category replaces the feature DTW considers. Derivative DTW [23] is a pioneer work in this category. It employs local derivative rather than the raw value, because the authors believe the trend of a data point is more important. For example, two points may have the same value, but they should be considered different if one is part of a ascending



(a) Sakoe-Chiba band (b) Itakura parallelogram (c) learned window

(d) step patterns

Fig. 3. Two types of constraints on the warping path: windowing constraint (a-b-c) and step pattern (d).

trend while the other is part of a descending trend. After Derivative DTW, many other types of features have also been considered, such as local maximum, local minimum [36], shape context [4,37], and interest-points [1], etc. It is also possible to combine different distance measures to create new methods, for example, combining Derivative DTW with DTW [16] or combining DTW with Euclidean distance [8].

These two categories follow different clues and they are independent with each other. The method we propose in this paper belongs to a new constraint for the warping path, so we concentrate on the methods of the first category in this paper.

3. LDTW : DTW with limited warping path length

3.1. Motivation

Windowing constraints limit the number of points each point can link to. It is an intuitive and straightforward solution to the pathological alignment problem. However, it takes the risk of preventing the correct alignment from being found due to its rigidity. Moreover, it is not easy to find an appropriate size or shape for the constraint window. Other types of constraints such as step patterns and slope weighting also suffer from the similar drawbacks.

In fact, another intuitive type of constraint is still open and seems promising. That is to limit the total number of links between two time series, rather than the number of links each point involves. This way, the pathological alignment will be alleviated because we prevent the tremendous amount of unreasonable links from happening by setting an upper bound to the total number of links. In other words, the number of links is not sufficient enough for singularities to form. In addition, we gain more flexibility because we still let the optimization procedure of DTW decide the number of points each point links to and where these links happen. We call DTW under this new constraint LDTW, because it actually requires us to solve DTW under Limited warping path length.

3.2. LDTW algorithm

The calculation of LDTW is not trivial. As already shown, DTW only concentrates on the minimum cumulative cost at each stage, regardless of the warping path length. However, LDTW puts a constraint on the warping path length. Therefore, we have to consider the current length of the warping path as an additional factor. The path length is decided by the number of steps that have been taken and in order to describe the algorithm more intuitively, we sometimes use the term “step

count". Note that $step\ count = path\ length - 1$ because path length takes into account the starting point. After considering the factor of step count (denoted by s), the problem of computing DTW under a fixed warping path length can be formulated recursively into Eq. (4). Compared with the formulation of DTW shown in Eq. (2), an extra dimension dedicated to record the current step count is added to the cumulative cost matrix.

$$D(A_r, B_c, s) = \delta(a_r, b_c) + min\begin{cases} D(A_r, B_{c-1}, s-1) \\ D(A_{r-1}, B_c, s-1) \\ D(A_{r-1}, B_{c-1}, s-1) \end{cases} \tag{4}$$

To solve Eq. (4), we rely on the similar dynamic programming principle used for DTW and the key point is to memorize the results of all subproblems. However, in DTW each cell inside the cumulative cost matrix only keeps one value, which is the global minimum cumulative cost among all possible path lengths. In contrast, in LDTW each cell has to keep multiple values, each of which is the minimum cumulative cost under a certain path length. For example, in the same cell at position (r, c), $m[r][c][s]$ is the result of subproblem $D(A_r, B_c, s)$ and $m[r][c][s-1]$ is the result of subproblem $D(A_r, B_c, s-1)$.

Note that Eq. (4) only defines how to calculate DTW under a fixed warping path length. But the goal of LDTW is to find the minimum DTW distance when the warping path length is shorter that a given upper bound L_{UB}. Therefore, a natural solution is to calculate the results for all admissible path lengths, and then choose the smallest one among those results. To achieve this solution, the first job is to ascertain the range of all admissible path lengths. According to the rules of DTW, the strict lower bound of the warping path length is $max(R, C)$ and we already have a given upper bound L_{UB}, so the range of path lengths is $(max(R, C), \ldots, L_{UB}]$. Next, we are supposed to calculate DTW under each admissible path length, but fortunately in fact we only need to calculate DTW when the path length is fixed at the upper bound L_{UB} if we exhaust all possible subproblems at each cell of the cumulative cost matrix (for example, $D(A_r, B_c, s)$ is a subproblem at cell (r, c)). Because of the recursive nature of these subproblems, as shown in Eq. (4), a subproblem under a longer path is based on subproblems under shorter paths. If we calculate DTW under each path length from scratch, we will encounter many overlap subproblems and eventually each subproblem will be encountered. So we decide to solve all intermediate subproblems once and for all. In this way, after we solve DTW under the longest path length L_{UB}, all subproblems required to solve DTW under shorter path lengths have already been calculated and we can get these results directly in the upper-right cell.

To exhaust all possible subproblems at each cell, the primary task is to ascertain the range of path lengths that each matrix cell has to consider. This is equivalent to computing the minimum and maximum steps allowed to reach a certain matrix cell. Given the rule of DTW, the minimum steps to reach a cell, for instance (r, c), is clear. It is just the larger one between $r-1$ and $c-1$, as shown in Algorithm 2. The red line in Fig. 4 illustrates one shortest path to reach (4, 3) with

Algorithm 2. Minimum steps to reach matrix cell (r, c).

Require: r row index of the cumulative cost matrix
Require: c column index of the cumulative cost matrix
1: **function** MIN_STEPS(r, c)
2: **return** $max(r, c) - 1$
3: **end function**

three steps, and the blue line illustrates one longest path with four steps. Although a path with five steps is not explicitly forbidden given the rules of DTW, we will never pick such a path because it involves orthogonal steps (shown in Fig. 4 with the dashed line) that will always lead to an extra cost. As we have mentioned, passing a matrix cell will incur a cost, therefore two orthogonal steps always have a higher cost than a diagonal step.

The computing of maximum steps to reach a cell is a bit tricky. First let us discuss the natural situation where no limit is set on the warping path length. In this situation, if the cell is located in the bottom row ($r = 1$) or the left-most column ($c = 1$), then the maximum steps is also equal to the larger one between $r-1$ and $c-1$. Elsewhere, the maximum steps is equal to $(c-2) + 1 + (r-2) = r + c - 3$, for example, we first take $(c-2)$ horizontal steps to reach the last but one column, then a diagonal step to reach the last column, and finally $(r-2)$ vertical steps to reach the end. Algorithm 3 summarizes

Algorithm 3. Maximum steps to reach matrix cell (r, c) with no limit on the warping path length.

Require: r row index of the cumulative cost matrix
Require: c column index of the cumulative cost matrix
1: **function** MAX_STEPS_NO_LIMIT(r, c)
2: **if** $r == 1 or c == 1$ **then**
3: **return** $max(r, c) - 1$
4: **else**
5: **return** $r + c - 3$
6: **end if**
7: **end function**

the above process. Then let us discuss the situation where a upper bound L_{UB} on the warping path length is given. In this

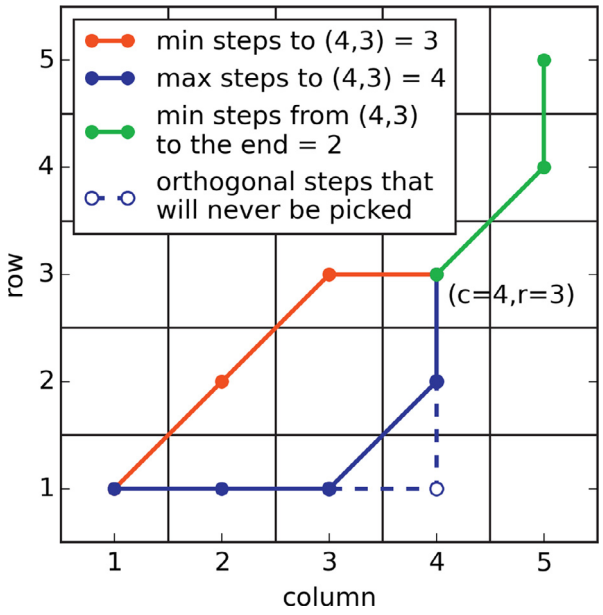


Fig. 4. Maximum and minimum steps allowed to reach a certain cell.

case, the total quota of steps is $L_{UB} - 1$. After reaching (r, c), we still need at least $max(R - r, C - c)$ steps to get to the end, as shown by the green line in Fig. 4. So we should reserve these steps in advance when we compute the maximum steps to reach (r, c). As a result, the current upper bound of maximum steps should be the total step quota $(L_{UB} - 1)$ minus the reserved steps and finally we choose the smaller one between the maximum steps in natural situation and the current upper bound. Algorithm 4 summarizes the complete procedure.

Algorithm 4. Maximum steps to reach matrix cell (r, c) given limited warping path length.

Require: r row index of the cumulative cost matrix
Require: c column index of the cumulative cost matrix
Require: L_{UB} the given upper bound of warping path length
Require: R length of the first time series
Require: C length of the second time series
1: **function** MAX_STEPS(r, c, L_{UB}, R, C)
2: $candidate_1 = max\_steps\_no\_limit(r, c)$
3: $candidate_2 = L_{UB} - 1 - max(R - r, C - c)$
4: **return** $min(candidate_1, candidate_2)$
5: **end function**

The next job is to fill the cumulative cost matrix according to Eq. (4). For each matrix cell, we compute the cumulative costs under different path lengths respectively. Fig. 5 illustrates how we consider different path lengths respectively. The numbers labeled on each cell show all possible step counts to reach that cell. To reach cell (5, 4), the number of steps ranges from 4 to 6. When computing the 5-step cost of that cell ($m[5][4][5]$), we only consider the 4-step costs of its left, lower-left, and lower neighbors. In parallel, when computing the 4-step cost, we only consider the 3-step costs of the three neighbor cells. Fig. 5 also shows another situation that is different from DTW. In DTW we always have to consider three adjacent cells at each step, but in LDTW some cells could be inaccessible due to limited step count. As shown in Fig. 5, we cannot get to cell (3, 1) with only one step, but $m[3][2][2]$ only considers the 1-step cost of adjacent cells, so the cell (3, 1) has to be ignored in the computation of $m[3][2][2]$.

After the cumulative cost matrix is filled, the upper-right cell (R, C) contains those results to be compared. Each of them is an optimal cumulative cost when the warping path length is fixed at a different value that is smaller than the upper bound L_{UB}. For instance, $m[R][C][L - 1]$ is the minimum cumulative cost when the warping path length is fixed at L. We select the smallest value among these results to be the LDTW distance. Algorithm 5 summarizes the complete LDTW procedure.

The algorithms to backtrack the warping path generated by LDTW and DTW are almost identical, but we still need to notice the differences. For LDTW, the cumulative cost matrix has a third dimension to indicate the step count, so the actual step count must be provided as input, which means we should also record the final step count when calculating LDTW. For

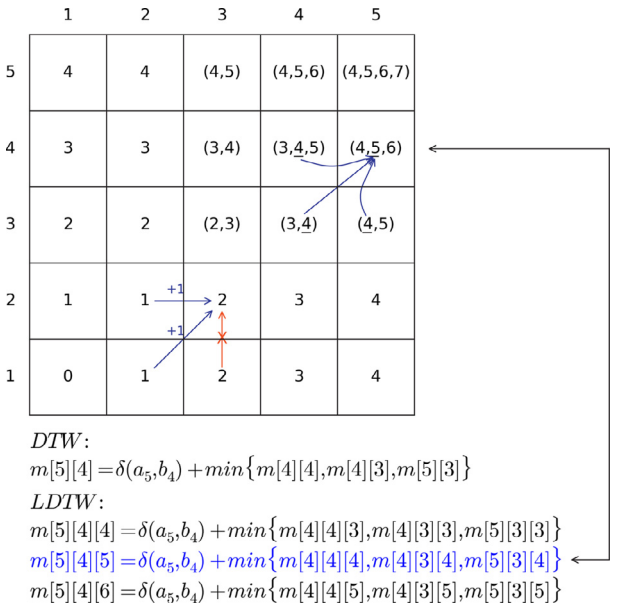


Fig. 5. LDTW considers different path lengths respectively. The numbers labeled on each cell are all possible step counts to reach that cell.

example, if $m[R][C][s]$ is the smallest value in the final cell (R, C), then s is the final step count. In addition, remember to decrease the current step count by one after each loop.

4. Experiment

The main purpose of DTW and its variants is to measure the similarity between numerical sequences, so in this section we evaluate the performance of LDTW as a distance measure by nearest neighbor classification experiment.[1]

4.1. Datasets

Experiments are conducted on twenty-two datasets from the UCR time series classification archive [9], the largest online time series database widely referenced by the community in recent years. One big advantage of the archive is its diversity. It consists of mostly real time series datasets collected from various domains and some synthetic ones. For example, Gun-Point dataset is collected from video trajectory recognition field; ECG200 dataset comes from electrocardiogram signals; Swedish Leaf dataset is converted from the contours of fifteen different leaf species; while Synthetic Control, CBF, and Two-Patterns datasets are synthetic. Table 2 summarizes the basic information of all datasets used in the experiments, such as the number of classes, size of training set, size of testing set and time series length. The number of classes ranges from 2 (e.g. Gun-Point, Wafer, etc) to 50 (50Words). The length of time series ranges from 60 (Synthetic Control) to 637 (Lightning-2).

4.2. 1NN accuracy report

One nearest neighbor classification (1NN for short) is strongly recommended [9,11] to demonstrate the performance of a distance measure, because 1NN has no parameters and thus the accuracy solely depends on the distance measure. In Table 3, we report the 1NN error rate of LDTW with the best upper bound of warping path length, which is learned from the training sets by cross validation and the details will be introduced in Section 4.6. We compare LDTW with four most widely used distance measures between time series : Euclidean distance, naive DTW, DTW under best Sakoe–Chiba band (Fig. 3(a)), and DTW under best Itakura parallelogram (Fig. 3(b)). These best parameters are also learned by cross validation on the training sets. From Table 3 we can observe that LDTW get the lowest error rates on 19 out of 22 datasets and the improvements are considerable especially on OSU Leaf (5%), 50Words (6%), Fish (7%), and Car (10%). On several datasets, DTW error rates are close to zero, but LDTW still achieves better accuracies. These results demonstrate the utility of the proposed LDTW.

Fig. 6 shows the pair-wise comparison of 1NN accuracy between LDTW and its four competitors. Each dot represents a dataset, whose x-value is the 1NN accuracy generated by the method in competition and y-value is the 1NN accuracy generated by LDTW. In such a setting, a dot located above the diagonal indicates LDTW outperforms its competitor on

[1] Related code of this paper is available at http://sites.univ-rennes2.fr/costel/corpetti/LDTW/LDTW_CODE.zip

Algorithm 5. Complete LDTW procedure.

Require: $A = \langle a_1 \ldots a_r \ldots a_R \rangle$
Require: $B = \langle b_1 \ldots b_c \ldots b_C \rangle$
Require: L_{UB} the given upper bound of warping path length
Let $\delta()$ be the cost between two points of different time series
Let $m[r][c][s]$ be the cumulative cost matrix
Let L be the actual length of the warping path

function LDTW(A, B, L_{UB})
  ▷ *Initializing the first column and row of m*
  $m[1][1][0] = \delta(a_1, b_1)$
  for $r = 2$ *to* R **do**
    $m[r][1][r-1] = m[r-1][1][r-2] + \delta(a_r, b_1)$
  end for
  for $c = 2$ *to* C **do**
    $m[1][c][c-1] = m[1][c-1][c-2] + \delta(a_1, b_c)$
  end for
  ▷ *Exhausting all subproblems at each cell of m*
  for $r = 2$ *to* R **do**
    for $c = 2$ *to* C **do**
      $min\_s = min\_steps(r, c)$
      $max\_s = max\_steps(r, c, L_{UB}, R, C)$
      for $s = min\_stomax\_s$ **do**
        $$m[r, c, s] = \delta(a_r, b_c) + min \begin{cases} m[\quad r \quad, c-1, s-1] \\ m[r-1, \quad c \quad, s-1] \\ m[r-1, c-1, s-1] \end{cases}$$
      end for
    end for
  end for
  ▷ *Selecting the best result and the corresponding path length in the last cell* (R, C)
  $min\_s\_final = max(R, C)$
  $max\_s\_final = L_{UB} - 1$
  $ldtw = +\infty$
  $L = 0$
  for $s = min\_s\_finaltomax\_s\_final$ **do**
    if $m[R][C][s] < ldtw$ **then**
      $ldtw = m[R][C][s]$
      $L = s + 1$
    end if
  end for
  return $ldtw$
end function

that dataset and vice versa. A dot laying on the diagonal means a tie and the farther a dot is above the diagonal, the better LDTW performs. The number of dots above, on, and below the diagonal are depicted on each graph. For instance on Fig. 6(b), LDTW outperforms naive DTW distance on 15 out of 22 datasets, ties on 6 datasets and loses on only 1 dataset. These graphs visually confirm the high superiority of LDTW on this variety of datasets.

4.3. Statistical significance test

To prove that the accuracy improvement made by LDTW is statistical significant, we first conducted omnibus Friedman test with Iman and Davenport's correction [13,18] to detect if at least one of the methods performed significantly different from the others. And then we applied pair-wise Bergmann and Hommel test [5,15] to compare methods. In both tests, we used the accuracy data shown in Table 3 and the R package "scmamp".

We passed the omnibus test with $chi - squared = 12.617$ and $p - value = 4.388e - 08 < 0.05$. Then Tables 4 and 5 list the raw and corrected results in the Bergmann and Hommel test, where each $p - value$ represents the possibility that the two methods being compared are *not* significantly different. The smaller the $p - value$ is, the more differently two methods perform. If $p - value < 0.05$, we believe they are *significantly* different.

Pairwise comparisons of methods show that all observed differences between LDTW and well-established methods are statistically significant, except for IP-DTW. But the $p - value$ of LDTW versus IP-DTW is still relatively small (raw: $0.057 < 0.1$, corrected: $0.113 < 0.15$). And judging from IP-DTW's point of view, it is not significantly different from LDTW,

Table 2
Basic information of datasets.

Dataset	Number of classes	Size of training set	Size of testing set	Time series length
Synthetic Control	6	300	300	60
Gun-Point	2	50	150	150
CBF	3	30	900	128
Face(all)	14	560	1690	131
OSU Leaf	6	200	242	427
Swedish Leaf	15	500	625	128
50Words	50	450	455	270
Trace	4	100	100	275
Two Patterns	4	1000	4000	128
Wafer	2	1000	6164	152
Face(four)	4	24	88	350
Lightning-2	2	60	61	637
Lightning-7	7	70	73	319
ECG200	2	100	100	96
Adiac	37	390	391	176
Yoga	2	300	3000	426
Fish	7	175	175	463
Plane	7	105	105	144
Car	4	60	60	577
Beef	5	30	30	470
Coffee	2	28	28	286
OliveOil	4	30	30	570

Table 3
Comparisons on 1NN error rate.

Dataset	1NN Euclidean distance	1NN DTW	1NN DTW (best Sakoe–Chiba band (r%)[a])	1NN DTW (best Itakura parallelogram (d)[b])	1NN LDTW (best upper bound of warping path length)	Cross validation[c] with LDTW (step size)
Synthetic Control	0.120	**0.007**	0.017 (6)	**0.007 (9)**	**0.007 (72)**	71–74 (1)
Gun-Point	0.087	0.093	0.087 (0)	0.027 (5)	**0.020 (158)**	156–161 (1)
CBF	0.148	**0.003**	0.004 (11)	0.004 (55)	**0.003 (198)**	142–254 (1)
Face(all)	0.286	0.192	0.192 (3)	**0.179 (8)**	0.198 (147)	147 (5)
OSU Leaf	0.479	0.409	0.388 (7)	0.355 (50)	**0.301 (468)**	468 (10)
Swedish Leaf	0.211	0.208	0.154 (2)	0.150 (6)	**0.131 (134)**	134 (1)
50Words	0.369	0.310	0.242 (6)	0.255 (29)	**0.180 (301)**	301 (10)
Trace	0.240	**0.000**	0.010 (3)	0.010 (7)	**0.000 (416)**	286–546 (10)
Two Patterns	0.090	**0.000**	0.002 (4)	0.00025 (10)	**0.000 (194)**	139–254 (5)
Wafer	0.005	0.020	0.005 (1)	0.005 (3)	**0.003 (155)**	154–157 (1)
Face(four)	0.216	0.170	0.114 (2)	0.170 (14)	**0.102 (386)**	386 (5)
Lightning-2	0.246	0.131	0.131 (6)	0.131 (35)	**0.098 (778)**	748–818 (10)
Lightning-7	0.425	0.274	0.288 (5)	0.247 (24)	**0.205 (355)**	335–380 (5)
ECG200	**0.120**	0.230	**0.120 (0)**	**0.120 (1)**	**0.120 (98)**	97–99 (1)
Adiac	0.389	0.396	0.391 (3)	0.376 (15)	**0.338 (177)**	177 (5)
Yoga	0.170	0.164	0.155 (2)	**0.135 (8)**	0.149 (467)	447–487 (10)
Fish	0.217	0.177	0.154 (4)	0.206 (9)	**0.086 (474)**	474 (10)
Plane	0.038	**0.000**	**0.000 (6)**	**0.000 (5)**	**0.000 (218)**	151–286 (1)
Car	0.267	0.267	0.233 (1)	0.283 (1)	**0.133 (598)**	598 (10)
Beef	0.333	0.367	0.333 (0)	**0.267 (12)**	0.333 (471)	471 (10)
Coffee	**0.000**	**0.000**	**0.000 (0)**	**0.000 (1)**	**0.000 (307)**	287–327 (10)
OliveOil	**0.133**	0.167	**0.133 (0)**	**0.133 (1)**	**0.133 (581)**	571–601 (10)

[a] r is the percentage of time series length.
[b] d is the half length of the shorter diagonal of the parallelogram.
[c] this column shows the optimal value or optimal zone of the upper bound of warping path length found by leave-one-out cross validation.

Table 4
Raw $p-value$ in Bergmann and Hommel test.

	Euclidean	DTW	SC-DTW	IP-DTW	LDTW
Euclidean	N/A	0.095205	0.006580	0.000419	5.49e-08
DTW	0.095205	N/A	0.294266	0.062991	0.000166
SC-DTW	0.006580	0.294266	N/A	0.417685	0.006580
IP-DTW	0.000419	0.062991	0.417685	N/A	0.056530
LDTW	5.49e-08	0.000166	0.006580	0.056530	N/A

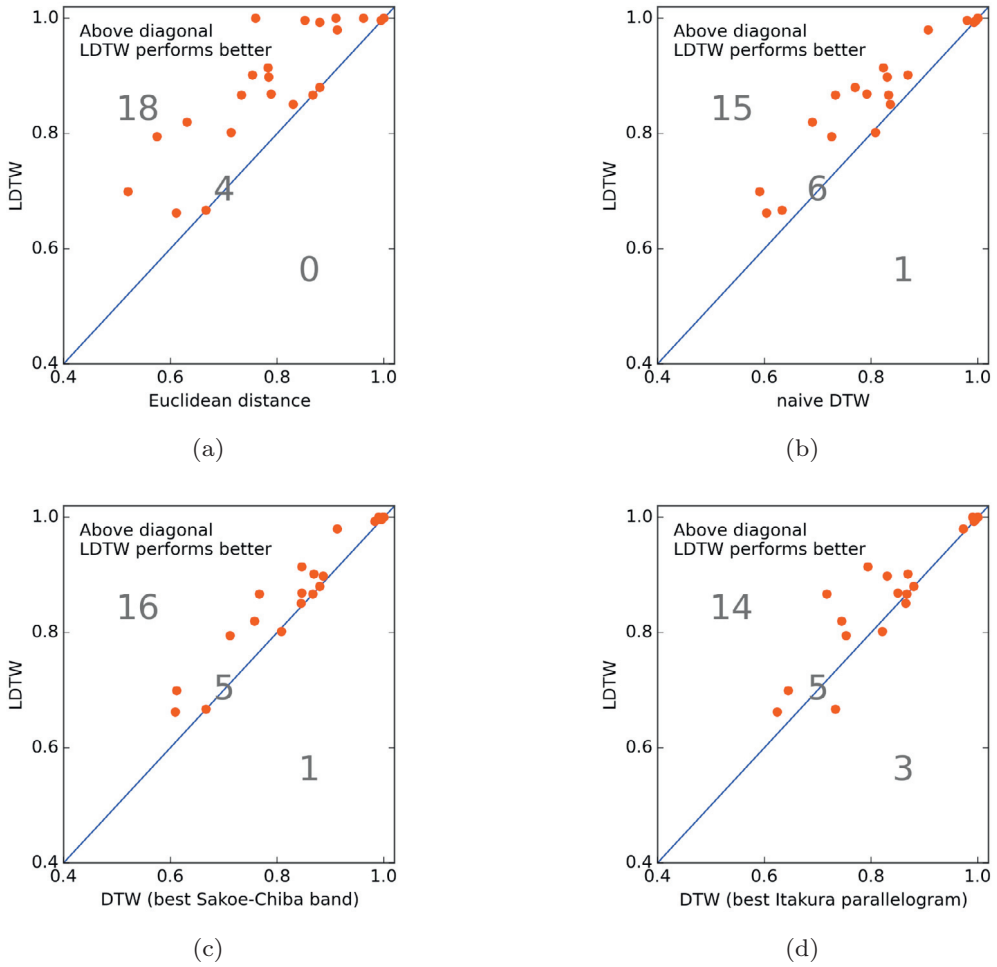


Fig. 6. 1NN accuracy of LDTW versus the four competitors respectively.

Table 5
Corrected $p-value$ in Bergmann and Hommel test.

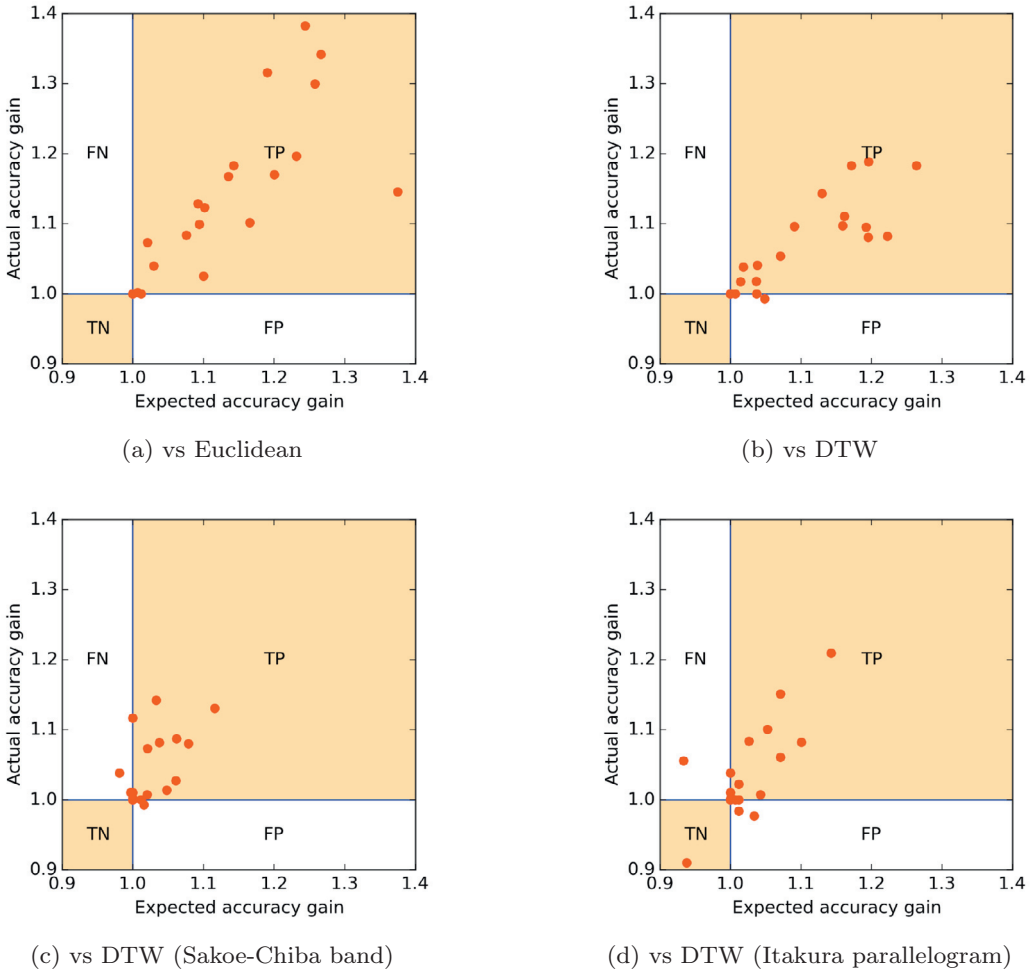
	Euclidean	DTW	SC-DTW	IP-DTW	LDTW
Euclidean	N/A	0.190409	0.026321	0.002514	5.49e-07
DTW	0.190409	N/A	0.294266	0.188974	0.000995
SC-DTW	0.026321	0.294266	N/A	0.417685	0.026321
IP-DTW	0.002514	0.188974	0.417685	N/A	0.113061
LDTW	5.49e-07	0.000995	0.026321	0.113061	N/A

SC-DTW, and even DTW. These results provide us statistical evidence to claim that LDTW outperforms those well-established methods.

4.4. The texas sharpshooter fallacy

A simple but pervasive logic error in time series classification papers is that they claim a method is useful only because the method has better accuracy on some datasets. However, this is not very useful unless we can tell ahead of time that on which datasets it will be more accurate [3]. To avoid this logic error, we must prove that we can predict in advance when LDTW will have superior accuracy. We make the prediction by measuring the *expected* accuracy gain *looking only at the training data* as:

$$gain = \frac{accuracy\ \ LDTW}{accuracy\ \ competitor} \tag{5}$$



(a) vs Euclidean

(b) vs DTW

(c) vs DTW (Sakoe-Chiba band)

(d) vs DTW (Itakura parallelogram)

Fig. 7. Expected accuracy gain calculated on training data versus actual accuracy gain.

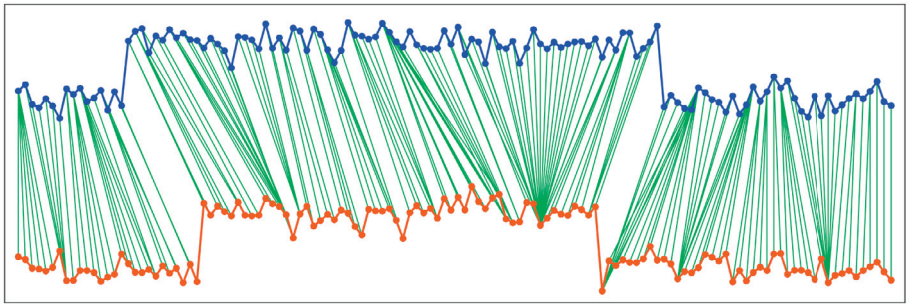
On training set, the accuracy is calculated by leave-one-out cross validation. If a parameter is involved, we choose the one with best accuracy. Gain values larger than one indicate that we expect LDTW will outperform the competitor on a given dataset and vice versa. We also calculated the *actual* accuracy gain using testing data.

To testify our prediction, we compare the *expected* accuracy gain with the *actual* accuracy gain. Fig. 7 shows the results of LDTW versus different competitors respectively. Each point represents a dataset. The comparison results in four cases:

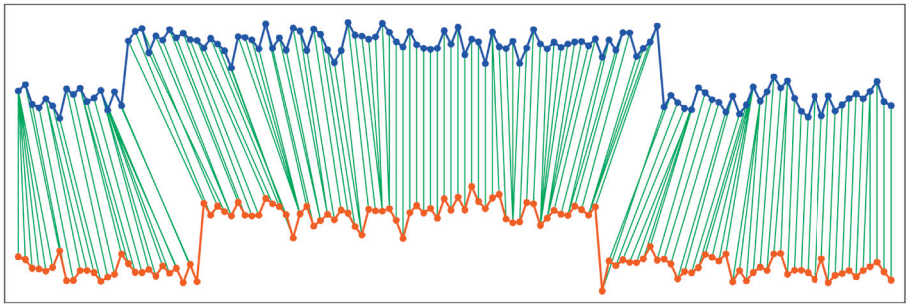
- TP(True positive): In this region we predicted that LDTW would improve accuracy and we were correct. This is the most beneficial situation for LDTW. Most points fall into this region, which demonstrates the real utility of LDTW.
- TN(True negative): In this region we correctly predicted that LDTW would decrease accuracy. In this case, we can avoid the loss of accuracy by choosing the other method.
- FN(False negative): In this region we predicted that LDTW would decrease accuracy, but the accuracy actually improved.
- FP(False positive): In this region we predicted that LDTW would improve accuracy, but the accuracy actually decreased. This is the only truly bad case. But only few points are in this region.

4.5. A close look at the alignment generated by LDTW

One primary motivation of LDTW is to solve the pathological alignment problem by limiting the total number of links between two time series. To visually demonstrate that LDTW can achieve that goal, let us take a close look at the alignment generated by DTW and LDTW in Fig. 8. We can observe that DTW does lead to several singularities, but they are sufficiently suppressed by LDTW. The limited number of links in LDTW are allocated more evenly while capturing those significant time shifts.

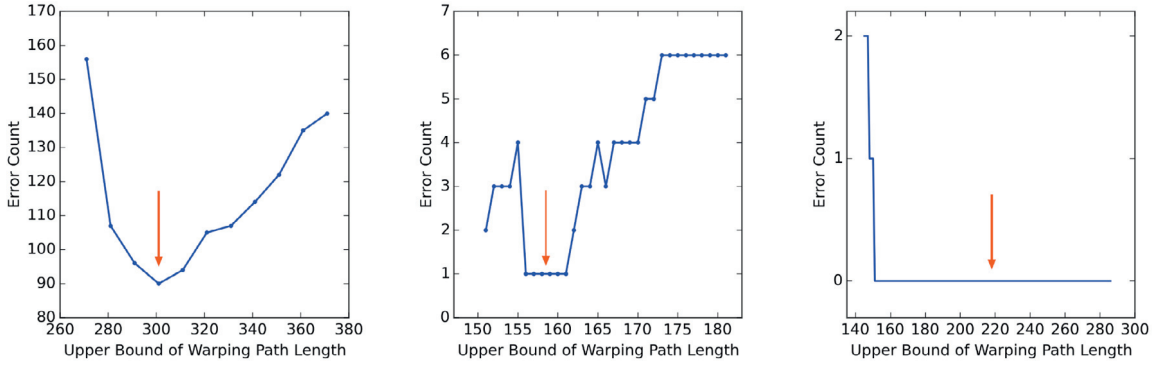


(a) alignment generated by DTW



(b) alignment generated by LDTW (warping path length = 149)

Fig. 8. Comparison between alignments generated by DTW and LDTW. The two time series here are the first two in the CBF training set.



(a) 50Words dataset (b) Gun Point dataset (c) Plane dateset

Fig. 9. Three typical error count curves in LOOCV (leave-one-out cross validation) processes.

4.6. Extracting reliable upper bound of warping path length

The only parameter of LDTW is the upper bound of warping path length L_{UB}. The effective range of L_{UB} is finite under the rule of DTW. If the lengths of two time series are R and C, then $max(R, C) + 1 \leq L_{UB} \leq R + C - 2$. In our experiment, taking the 50Words dataset as an example, the time series length is 270, so the search space of L_{UB} is $[271,\ 538(270 + 270 - 2)]$. Based on the search space, we can learn an appropriate L_{UB} by LOOCV (leave-one-out cross validation) on the training set. Sometimes the search space may be too large, and in that case we can set a larger step size instead of testing every candidates. The step size we use for each dataset is shown within the parentheses in the last column of Table 3, and they are roughly proportional to the length of time series.

During LOOCV, each candidate results in a 1NN classification error count. Fig. 9 shows the LOOCV processes on three datasets and they exhibit two typical situations. Fig. 9(a) belongs to the first situation where we can observe a single optimal candidate that corresponds to the minimum error count, and in this case we choose this candidate directly. Fig. 9(b) belongs to the second situation where we can observe a continuous zone of candidates that all lead to relatively small error counts. The shapes of the error count curves in Fig. 9(a-b) are similar: the error gradually descends until it reaches the minimum

Table 6
1NN error rates of selecting minimum, maximum, and median/mean of optimal zone.

Dataset	Optimal zone	Minimum	Maximum	Median/Mean
Synthetic control	71–74	0.0100	**0.0067**	**0.0067**
Gun-Point	156–161	0.0267	**0.0200**	**0.0200**
CBF	142–254	**0.0022**	0.0033	0.0033
Trace	286–546	0.0100	**0.0000**	**0.0000**
Two patterns	139–254	0.0000	0.0000	0.0000
Wafer	154–157	0.0032	0.0034	**0.0028**
Lightning-2	748–818	0.0983	0.0983	0.0983
Lightning-7	335–380	0.2740	0.2603	**0.2055**
ECG	97–99	0.1200	**0.1100**	0.1200
Yoga	447–487	**0.1423**	0.1483	0.1490
Plane	151–286	0.0000	0.0000	0.0000
Coffee	287–327	**0.0000**	0.0357	**0.0000**
OliveOil	571–601	**0.1333**	0.1667	**0.1333**

point or minimum zone, and then it ascends until the end of the search space. Based on this observation, we decide to choose the median of the optimal zone as our optimal result. Note that if the size of the optimal zone is an even number, there will be two medians, and we choose the smaller one. Fig. 9(c) is a special case of the second situation, where the optimal zone happens at the end of the search space so there is no ascending phase. We still choose the median if the optimal zone happens at the end or the start. The last column of Table 3 lists the optimal value or optimal zone of L for each dataset.

To further demonstrate that selecting the median of optimal zone is fair, we compared the 1NN error rates when selecting minimum, maximum, and median/mean. The step size does not change during the entire LOOCV process and the best candidate must be an integer. In this setting, the mean is always equivalent to the median. For example, in [1, 3, 5, 7, 9], the median is 5 and the mean is also 5. Or if the number of elements is even, for example, in [1, 2, 3, 4, 5, 6], the smaller median is 3 and the integer part of its mean ($floor(3.5)$) is also 3. Table 6 lists the 1NN error rates of selecting minimum, maximum, or median/mean on every dataset involves an optimal zone. After a simple comparison, we can conclude that selecting the median is a better choice.

4.7. Contributions of LDTW

Apart from the interesting performances of LDTW in terms of classification, we also would like to point out the theoretical contributions of this technique.

First, although LDTW and some other existing variants of DTW, for instance, windowing constraint, both set constraints on the warping path, LDTW prefers a global one rather than local ones. As already shown in Section 2.3, given a typical windowing constraint such as Sakoe–Chiba band or Itakura parallelogram, a point can only link to the points temporally close enough to it, so windowing constraint essentially limits the local behavior of the warping path. Other variants of DTW such as step pattern and slope weighting also set constraints locally. In contrast, LDTW limits the total length of the warping path and this obviously belongs to a global constraint. As shown by the experimental results, LDTW demonstrates that global constraints may be more efficient than local constraints. We believe LDTW opens up a new direction of imposing different types of global constraints on DTW. It may take more advantages of the intrinsic optimization process of DTW and be more flexible.

Second, LDTW makes us follow a similar pattern of time distortions when comparing different pairs of time series. Excluding noises, the dissimilarities between time series mainly come from two sources: time distortions and intrinsic differences. The ideal distance measure only captures intrinsic dissimilarities while fully eliminates the dissimilarities caused by time distortions. But the definition of DTW implies that DTW tends to attribute all dissimilarities between time series to time distortions. If a set of time series come from the same sensor or are issued from the same underlying mechanism, there is a high possibility that they share a common pattern of time distortions. The extracted warping path reflects the pattern of time distortions among time series. To some degree, limiting or even fixing the length of warping path ensure the distance measure obeys this common pattern of time distortions, and thus make comparisons between sequences mainly affected by intrinsic dissimilarities rather than time distortions.

4.8. Time complexity and parallelization

In this subsection, we discuss the time complexity and running time of LDTW in comparison with other well-established methods. Let R and C be the lengths of two time series, and L_{UB} be the parameter (upper bound of warping path length) in LDTW. Euclidean distance can only deal with time series with the same length, and its time complexity is $\Theta(R \text{ or } C), R = C$. As DTW has to consider all cells in the warping matrix, its time complexity is $\Theta(R \cdot C)$. SC-DTW and IP-DTW set constraints on the warping path, so they only need to consider part of the warping matrix and their time complexities are both $O(R \cdot C)$.

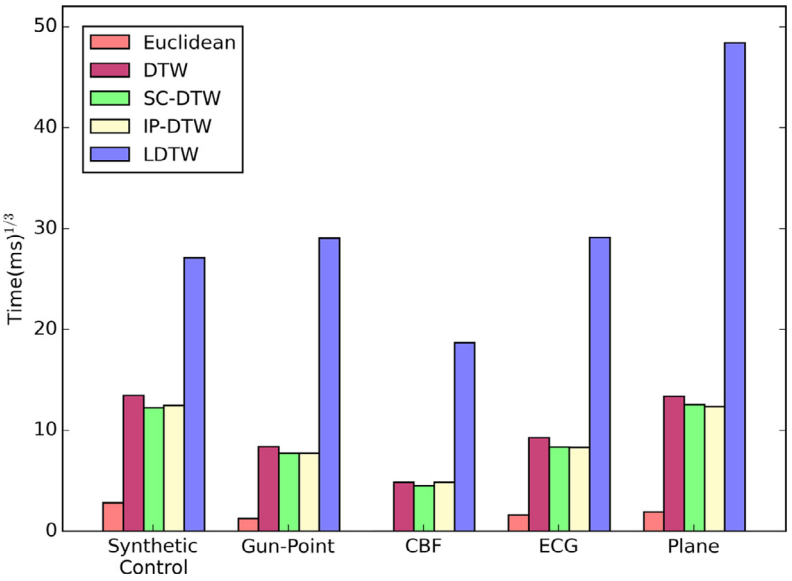


Fig. 10. Running time of the leave-one-out cross validation process on five different datasets.

Similarly to DTW, LDTW considers the entire warping matrix. However, for each matrix cell, LDTW also considers situations under different path lengths. As a result, LDTW has a time complexity of $O(L_{UB} \cdot R \cdot C)$.

To compare running time of different methods, we repeated twenty times the leave-one-out cross validation process on five different datasets, and Fig. 10 shows the average running time. We can see that the running time basically coincide with their own time complexities. LDTW is more time-consuming than existing methods. But even when running time would be the bottleneck, we always have enough tools to deal with it, for example, parallel computing, GPGPU, FPGA, etc. In addition, it should be outlined that the structure of the LDTW algorithm is suitable for parallelization in both bottom and top level. In bottom level, for each matrix cell, we can consider situations with different step counts in parallel. In top level, one can also compute LDTW between different pairs of time series in parallel. The classification of each time series is also independent and can be performed simultaneously.

5. Conclusion

In this paper, we have proposed a new variant of DTW, entitled LDTW, which limits the length of the warping path in DTW. LDTW sets a global and softer constraint instead of those local and rigid ones used by many existing variants of DTW. LDTW thus can efficiently alleviate the pathological alignment while keeping the flexibility of DTW.

To implement LDTW, we have extended the original DTW algorithm by adding a step count dimension to the cumulative cost matrix. All details associated with the LDTW algorithm have been introduced in this paper. To demonstrate the performance of LDTW, we have conducted a series of one nearest neighbor classification experiments on the newest UCR time series archive. Comparing with other most popular time series distance measures, LDTW shows better accuracy on most datasets. We therefore believe this work opens up a new direction of imposing global constraints on DTW instead local ones.

Acknowledgments

The authors would like to thank Prof. Eamonn Keogh and all the people who have contributed to the UCR time series classification archive for their selfless work. We also thank the anonymous reviewers for their valuable advice.

This work has been supported by Major Project of High Resolution Earth Observation System of China (Grant No.03-Y20A04-9001-15/16), the CNES TOSCA-VEGIDAR Program, and CAS-CNRS Joint Doctoral Promotion Program.

References

[1] A. Bailly, S. Malinowski, R. Tavenard, T. Guyet, L. Chapel, Bag-of-Temporal-SIFT-Words for time series classification, ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data, 2015.

[2] I. Bartolini, P. Ciaccia, M. Patella, Warp: accurate retrieval of shapes using phase of fourier descriptors and time warping distance, IEEE Trans. Pattern Anal. Mach. Intell. 27 (1) (2005) 142–147.

[3] G.E. Batista, X. Wang, E.J. Keogh, A complexity-invariant distance measure for time series., in: Proceedings of SIAM International Conference on Data Mining, 11, SIAM, 2011, pp. 699–710.

[4] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Trans. Pattern Anal. Mach. Intell. 24 (4) (2002) 509–522.
[5] B. Bergmann, G. Hommel, Improvements of general multiple test procedures for redundant systems of hypotheses, in: Multiple Hypothesenprüfung/Multiple Hypotheses Testing, Springer, 1988, pp. 100–115.
[6] D.J. Berndt, J. Clifford, Using dynamic time warping to find patterns in time series., in: KDD Workshop, 10, Seattle, WA, 1994, pp. 359–370.
[7] K.S. Candan, R. Rossini, X. Wang, M.L. Sapino, sDTW: computing DTW distances using locally relevant constraints based on salient feature alignments, Proc. VLDB Endowment 5 (11) (2012) 1519–1530.
[8] Y. Chen, B. Hu, E. Keogh, G.E. Batista, Dtw-d: time series semi-supervised learning from a single example, in: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2013, pp. 383–391.
[9] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, G. Batista, The UCR Time Series Classification Archive, 2015. www.cs.ucr.edu/~eamonn/time_series_data/.
[10] T.H. Cormen, Introduction to Algorithms, MIT Press, 2009.
[11] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, E. Keogh, Querying and mining of time series data: experimental comparison of representations and distance measures, Proc. VLDB Endowment 1 (2) (2008) 1542–1552.
[12] P.F. Felzenszwalb, R. Zabih, Dynamic programming and graph algorithms in computer vision, IEEE Trans. Pattern Anal. Mach. Intell. 33 (4) (2011) 721–740.
[13] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, J. Am. Stat. Assoc. 32 (200) (1937) 675–701.
[14] T.-c. Fu, A review on time series data mining, Eng. Appl. Artif. Intell. 24 (1) (2011) 164–181.
[15] S. Garcia, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, J. Mach. Learn. Res. 9 (2008) 2677–2694.
[16] T. Górecki, M. Łuczak, Using derivatives in time series classification, Data Min. Knowl. Discov. 26 (2) (2013) 310–331.
[17] B. Huang, W. Kinsner, ECG frame classification using dynamic time warping, in: IEEE Canadian Conference on Electrical and Computer Engineering, 2002, 2, IEEE, 2002, pp. 1105–1110.
[18] R.L. Iman, J.M. Davenport, Approximations of the critical region of the friedman statistic, Commun. Stat.-Theory Methods 9 (6) (1980) 571–595.
[19] F. Itakura, Minimum prediction residual principle applied to speech recognition, IEEE Trans. Acoust. Speech Signal Process. 23 (1) (1975) 67–72.
[20] Y.-S. Jeong, M.K. Jeong, O.A. Omitaomu, Weighted dynamic time warping for time series classification, Pattern Recognit. 44 (9) (2011) 2231–2240.
[21] B. Kartikeyan, A. Sarkar, Shape description by time series, IEEE Trans. Pattern Anal. Mach. Intell. 11 (9) (1989) 977–984.
[22] E. Keogh, C.A. Ratanamahatana, Exact indexing of dynamic time warping, Knowl. Inf. Syst. 7 (3) (2005) 358–386.
[23] E.J. Keogh, M.J. Pazzani, Derivative dynamic time warping., in: Proceedings of SIAM International Conference on Data Mining, 1, SIAM, 2001, pp. 5–7.
[24] R. Kohavi, et al., A study of cross-validation and bootstrap for accuracy estimation and model selection, in: Ijcai, 14, 1995, pp. 1137–1145.
[25] J. Kruskall, M. Liberman, The symmetric time warping algorithm: From continuous to discrete, Time Warps, String Edits and Macromolecules, 1983.
[26] T.W. Liao, Clustering of time series data - a survey, Pattern Recognit. 38 (11) (2005) 1857–1874.
[27] C. Myers, L.R. Rabiner, A.E. Rosenberg, Performance tradeoffs in dynamic time warping algorithms for isolated word recognition, IEEE Trans. Acoust. 28 (6) (1980) 623–635.
[28] F. Petitjean, G. Forestier, G. Webb, A.E. Nicholson, Y. Chen, E. Keogh, et al., Dynamic Time Warping averaging of time series allows faster and more accurate classification, in: IEEE International Conference on Data Mining (ICDM), 2014, IEEE, 2014, pp. 470–479.
[29] F. Petitjean, J. Inglada, P. Gançarski, Satellite image time series analysis under time warping, IEEE Trans. Geosci. Remote Sens. 50 (8) (2012) 3081–3095.
[30] L.R. Rabiner, A.E. Rosenberg, S.E. Levinson, Considerations in dynamic time warping algorithms for discrete word recognition, IEEE Trans. Acoust. 26 (6) (1978) 575–582.
[31] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, E. Keogh, 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, 2012, pp. 262–270.
[32] C.A. Ratanamahatana, E. Keogh, Making time-series classification more accurate using learned constraints, in: Proceedings of SIAM International Conference on Data Mining, Lake Buena Vista, Florida, 2004, pp. 11–22.
[33] H. Sakoe, S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, IEEE Trans. Acoust. Speech Signal Process. 26 (1) (1978) 43–49.
[34] C. Tappert, S. Das, Memory and time improvements in a dynamic programming algorithm for matching speech patterns, IEEE Trans. Acoust. 26 (6) (1978) 583–586.
[35] D. Yu, X. Yu, Q. Hu, J. Liu, A. Wu, Dynamic time warping constraint learning for large margin nearest neighbor classification, Inf. Sci. 181 (13) (2011) 2787–2796.
[36] Z. Zhang, L. Tang, P. Tang, Local feature based dynamic time warping, in: IEEE International Conference on Data Science and Advanced Analytics (DSAA), 2014, IEEE, 2014, pp. 425–429.
[37] Z. Zhang, P. Tang, R. Duan, Dynamic time warping under pointwise shape context, Inf. Sci. 315 (2015) 88–101.