第49卷第2期 中国科学技术大学学报 Vol. 49, No. 2
2019年2月 JOURNAL OF UNIVERSITY OF SCIENCE AND TECHNOLOGY OF CHINA Feb. 2019

文章编号:0253-2778(2019)02-0138-11

基于趋势信息的时间序列分类方法

林钱洪[1,2],王志海[1,2],原继东[1,2],张 伟[1,2]

(1.北京交通大学计算机与信息技术学院 北京 100044;2.交通数据分析与挖掘实验室(北京交通大学),北京 100044)

摘要:大部分时间序列数据分析的一个重要组成部分是相似性度量方式.在众多相似性度量方式中,基于最长公共子序列的相似性度量方式是一种常用的有效方法,但该方法仅仅度量序列点对点的数值差异,而忽略了序列的变化趋势.为此提出一种基于趋势信息的时间序列离散化方法并用最长公共子序列进行相似性度量.该方法能够很好地度量时间序列的趋势信息.此外,还将其与现有的点对点函数线性结合.与现有相似性度量方法不同,该方法能同时考虑时间序列的趋势信息和函数距离,相似性度量方案运用最近邻分类算法规则进行分类.为了进行全面的比较,在42个时间序列数据集上测试该算法的有效性.实验结果表明,所提出的方法能有效提高时间序列分类准确率.

关键词:时间序列;趋势信息;时间序列离散化;相似性度量

中图分类号:TP391 **文献标识码**:A doi:10.3969/j.issn.0253-2778.2019.02.009

引用格式:林钱洪,王志海,原继东,等.基于趋势信息的时间序列分类方法[J].中国科学技术大学学报,2019,49(2):138-148.

LIN Qianhong, WANG Zhihai, YUAN Jidong, et al. Trend information for time series classification [J]. Journal of University of Science and Technology of China, 2019, 49(2): 138-148.

Trend information for time series classification

LIN Qianhong[1,2], WANG Zhihai[1,2], YUAN Jidong[1,2], ZHANG Wei[1,2]

(1. *School of Computer and Information Technology, Beijing Jiaotong University, Beijing* 100044;
2. *Laboratory of Traffic Data Analysis and Mining (Beijing Jiaotong University), Beijing* 100044)

Abstract: One of most important parts of time series data analysis is to choose the appropriate similarity measurement. Among all similarity measurements, the longest common subsequence is a commonly used and effective method. However, the original method only measures the numerical differences of point-to-point sequences, which neglects the trend of the changing sequence. Therefore, a time series discretization method based on the trend information is proposed and the longest common subsequence is employed to carry out similarity measurements. This method can measure time series trend information well. In addition, it is linearly combined with the point-to-point comparison function. In contrast to well-known measures from the literature, the proposed method can take both the trend information of time series and point-to-point comparison function into consideration. The new similarity measurement is used in classification with the nearest neighbor rule. In order to provide a comprehensive comparison, a set of experiments have been conducted, testing its effectiveness on 42 real time series. The experimental results

收稿日期:2018-07-17;**修回日期**:2018-09-18

基金项目:国家自然科学基金(61672086, 61702030, 61771058);北京市自然科学基金(4182052);中央高校基本科研业务费专项资金(2017YJS036)资助.

作者简介:林钱洪,男,1996生,硕士生.研究方向:机器学习,数据挖掘,时间序列分类.Email:17120380@bjtu.edu.cn

通讯作者:王志海,博士/教授.E-mail:zhhwang@bjtu.edu.cn

show that our method can effectively improve the accuracy rate of time series classification.
Key words: Time series; trend information; time series discretization; similarity measure

0　引言

时间序列数据存在于生活中的各个领域. 例如,股票价格数据、天气数据、心电图数据等. 其数据类型是实值型,具有数据量大、数据维度高以及数据不断更新等特点. 时间序列分类问题与传统的分类问题的主要区别在于,时间序列数据的各变量之间具有次序关系,而传统分类问题认为属性次序是不重要的,并且变量之间的相互关系独立于它们的相对位置,因此时间序列分类问题已成为数据挖掘领域的特殊挑战之一[1].

在时间序列分类算法中,最近邻算法(one-nearest neighbor, 1NN)是使用最多的分类算法之一. 该算法简单,在时间序列分类问题上往往能起到很好的分类效果. 最近邻算法中最重要的部分就是相似度衡量公式,因此时间序列间的相似度度量也成为研究人员的研究重点之一. 到目前为止研究人员已经提出了很多时间序列的相似度度量方法. 其中欧几里得距离作为一个无参数的距离度量公式,在许多时间序列问题上有很好的应用[2];为了解决时间序列在时间维度上的相位偏移问题,Ratanamahatana 等[3]提出动态时间规整算法(dynamic time warping, DTW),Latecki 等[4]提出最长公共子序列(longest common subsequence, LCS)算法. DTW 和 LCS 都是基于动态规划的思想计算序列间的弹性距离,可以有效地计算时间维度中某些非线性变化的相似性不变量,随后有研究人员对该方法进行改进. 例如,为了避免 DTW 方法种匹配路径过于扭曲,Jeong 等[5]提出了加权动态时间规整(weighted dynamic time warping, WDTW)算法;Kate[6]提出将 DTW 距离作为特征,再利用机器学习算法进行分类预测;Song 等[7]提出了窗口链模式的最长公共子序列算法(window-chained longest common subsequence, WCLCS),能够有效解决传统 LCS 算法中两序列比配单元距离较远的问题. Marteau 等[8]提出的时间规整编辑(time warp edit, TWE)结合了 DTW 和 LCS 的特点. 除了直接对原始算法加以改进之外,不少研究人员还对原始序列进行转化,形成新的时间序列相似性度量公式. 例如,Keogh 等[9]提出的基于导数的动态时间规整(derivative dynamic time warping, DDTW)算法;Górecki 等[10]提出了利用一阶导数和二阶导数信息的最长公共子序列算法用于时间序列数据分类;Schäfer 等[11]提出了 BOSS(bag of sfa symbols)算法,该方法利用傅里叶变换转换时间序列并用 MCB(multiple coefficient binning)方法对其离散化,使用特定的距离衡量公式进行时间序列相似性度量. 上述时间序列相似性度量算法中,大部分都是度量两序列点对点之间的函数距离. 在某些领域,点对点之间的函数距离并不能很好地度量两序列间的相似程度,相反序列的变化趋势和形状更为重要. 以往的基于导数的时间序列转化,虽然能够有效表示序列的形状信息,但是其受序列值影响较大,在一定情况下会影响分类效果. 本文提出利用序列的斜率信息来表示序列的变化趋势和形状[12]. 此外,为了避免受到原始序列值的影响,我们将序列的斜率进行离散化表示,仅保留原始序列的变化趋势信息. 我们用带约束的 LCS 相似度度量方法对离散化后的序列之间的相似性进行度量. 图 1 展示了时间序列转化之前和转化之后使用 LCS 进行相似度衡量,即两序列匹配点数目.

从图 1(左)可以看出,使用 LCS 直接对原始序列进行相似性度量时,序列间的匹配点存在偏差,最终成功匹配的点较少;图 1(右)展示了利用趋势信息对序列进行转化之后,再利用 LCS 进行相似性度量时序列间点匹配情况. 可以看出本文提出的方法对序列进行转化之后序列间匹配点更加准确,得到的匹配点增加,能更加准确地衡量序列间的相似度.

本文的主要贡献如下:

(Ⅰ)提出了一种基于趋势信息的时间序列离散化方法,该方法可以有效地保留原始序列的形状与变化趋势信息,并用 LCS 度量转化后序列的相似性. 该方法对于那些侧重衡量序列变化趋势的数据集有较好的效果;

(Ⅱ)将时间序列的变化趋势相似性度量与点对点函数距离度量相结合,形成新的相似度度量方案,该方法既能保留原有方法的有优点,也能够表示时间序列的变化趋势信息;

(Ⅲ)用不同的数据集表明本文提出的基于趋势信息的时间序列表示方法以及与点对点函数距离度

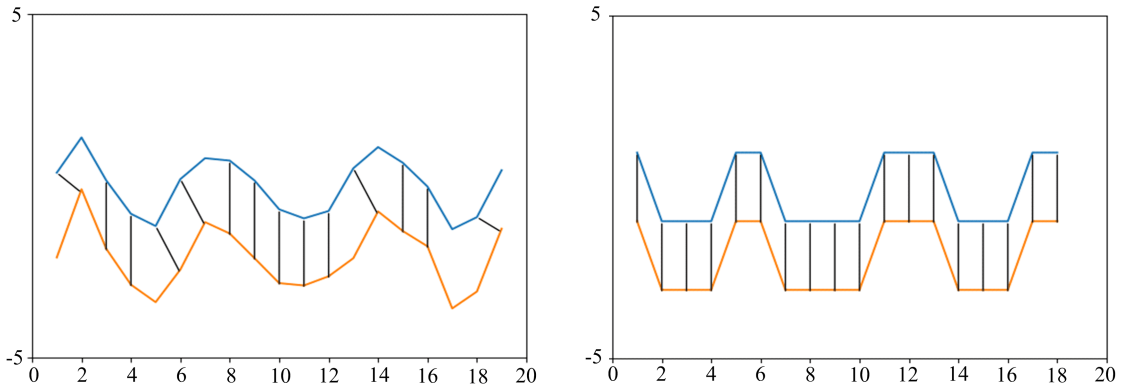

图 1 LCS 序列匹配

Fig. 1 LCS matching

量相结合的方法可以有效地对时间序列数据进行分类.

1 研究背景

本节将介绍最长公共子序列(LCS)算法和动态时间规整(DTW)算法的基本思想.

1.1 最长公共子序列

最长公共子序列的基本思想是通过两个序列进行匹配,而不重新排列元素的顺序,但允许某些元素不匹配或者遗漏;不同于欧几里得距离和 DTW 距离,必须使用序列中所有的点. 最终计算两序列中匹配点对的数目,但是一个点不能与另外一个序列中的两个或多个点进行匹配.

对于长度分别为 m 和 n 的序列 X 和 Y,给出其两序列的最长公共子序列的递归计算公式[10]为

$$L(i,j)=\begin{cases}0, i=0 \parallel j=0\\ 1+L(i-1,j-1), \mid i-j \mid \leqslant \delta,\\ \qquad\qquad \& \mid x_i-y_i \mid \leqslant \varepsilon\\ \max(L(i-1,j),L(i,j-1)), \text{others}\end{cases} \tag{1}$$

式中,δ 和 ε 为约束参数,δ 通常将其设置为序列长度的百分比,是一个相位偏移阈值,控制窗口大小以匹配给定点从一个序列到另一个序列中的点;$0<\varepsilon<1$ 是一个匹配阈值,如果点之间的距离小于 ε,则认为两个点匹配成功. 图 2 给出其示意图[10].

由公式(1)可以得出序列 X 和序列 Y 的最长公共子序列长度为 $L(m, n)$,给出与之对应的序列 X 和 Y 的相似性度量公式[13]为

$$\mathrm{LCSS}(x,y)=\frac{n+m-2L(n,m)}{n+m} \tag{2}$$

通过上述公式定义,则两个序列之间的相似性

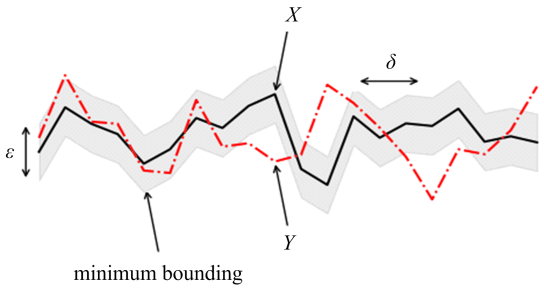

图 2 在时间 δ 和空间 ε 内可以匹配成功

Fig. 2 Matching within δ in time and ε in space

度量值的取值范围为$\left(1-2\frac{\min(n,m)}{n+m}\right)$到 1. 对于长度相等的两条序列,其值的取值范围为 0 到 1.

由上述最长公共子序列的算法可以看出,只有当同类时间序列数据的对应点之间的差值波动不大时,才能具有很好的匹配效果. 现实生活存在一些时间序列数据,同类型的序列之间变化趋势相同,但序列之间的点对点距离函数值较大. 这种情况下,原有的 LCS 不能很好地对时间序列数据进行分类.

1.2 动态时间规整

在时间序列分类和聚类算法中,DTW-1NN 算法是众多算法中最有效的方法之一. DTW 算法通过允许一个序列到另一个序列的非线性映射,并最小化两个序列之间的距离来找到两个序列之间的最佳匹配,能够有效地解决时间序列中存在的序列在时间维度上的相位偏移等问题. 下面给出 DTW 时间序列相似性度量方法[14].

假设长度为 m 的序列 $A=\{a_1, a_2, \cdots, a_m\}$ 和长度为 n 的序列 $B=\{b_1, b_2, \cdots, b_n\}$,我们用 $m*n$ 的矩阵 $\boldsymbol{D}$,其中元素 $\boldsymbol{D}[i][j]$ 表示 a_i 和 b_j 之间的距离. 则由 DTW 算法匹配的一条弯曲路径

$$P=((e_1,f_1),(e_2,f_2),\ldots,(e_s,f_s))$$

是矩阵 $\boldsymbol{D}$ 的一个遍历. 例如，当两序列长度相同时，欧氏距离的路径即是矩阵 $\boldsymbol{D}$ 的对角线. 弯曲路径 P 有如下约束：

（Ⅰ）端点约束路径起始点 $(e_1, f_1)=(1, 1)$ 且终点 $(e_s, f_s)=(m, n)$.

（Ⅱ）需要保持路径单调连续性，即

$0 \leqslant e_{i+1}-e_i \leqslant 1$ 且 $0 \leqslant f_{i+1}-f_i \leqslant 1$.

两个序列之间的最佳匹配是将一个序列与另一个序列对齐后具有最小距离路径的序列。在 DTW 中利用动态规划的思想递归求解，其递归求解公式[5]为

$$\left.\begin{aligned}\mathrm{DTW}_p(A,B)&=\sqrt[p]{d(i,j)}\\ d(i,j)&=|a_i-b_j|^p+\min\{d(i-1,j-1),\\ &\qquad d(i-1,j),d(i,j-1)\}\end{aligned}\right\}\tag{3}$$

式中，p 表示使用 p 范式，一般情况下我们选取 p 的值为 2.

由于时间序列数据维度较高，为了加快序列对齐速度以及提高准确率. 通常给上述算法添加一个窗口约束 w，以防止序列匹配相位偏移较大，在公式(3)中表示为 $|i-j| \leqslant w$. 窗口约束能加快 DTW 算法的序列对齐速度和提高分类准确率. 当给定两时间序列时，可以根据 DTW 算法在所约束的搜索空间中找出最短的弯曲路径. 图 3 给出了两时间序列 A 和 B 之间的弯曲矩阵和路径[5]，其中 w 为窗口大小.

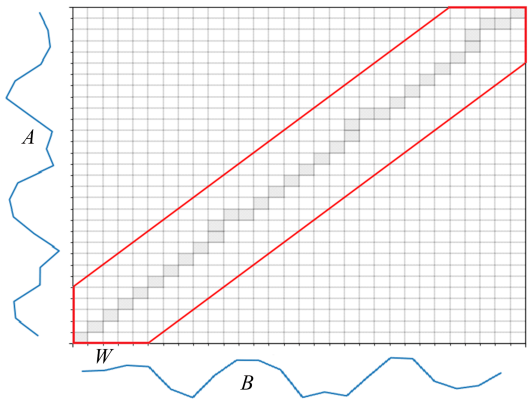

图 3　DTW 计算的两序列弯曲路径与矩阵

Fig. 3　Warping matrix and optional warping path by DTW

虽然 DTW 算法能很好地解决时间序列中的相位偏移问题，适用于绝大多数的时间序列数据，但其同样没能考虑序列的变化趋势. 我们将基于趋势信息的 LCS 与 DTW 相结合，以便于能同时考虑到序列的趋势信息和序列间点对点函数距离信息，更加充分地度量序列间的相似性.

2　改进方法

2.1　基于趋势信息的最长公共子序列(TLCS)

最长公共子序列直接对某些时间序列数据进行序列间的相似性度量时，往往不能取得很好的效果. 研究者们也通过很多途径对原始序列进行转换或对 LCS 进行改进，以便于得到更好的时间序列分类效果. 如利用 SAX(symbolic aggregate approximation)对序列进行转化，再用 LCS 对其进行分类；或在原始的 LCS 距离上添加基于导数的 LCS 距离为最终的距离度量公式[10]，但都没解决同类型序列之间对应点函数值差异较大的问题. 本文提出一种新的基于序列的趋势对原始序列进行离散化，再利用 LCS 对其进行相似性度量. 本文利用斜率来表示趋势信息，下面对该方法进行详细介绍.

给定序列 $X=\{x_1, x_2, \cdots, x_n\}$ 和 $Y=\{y_1, y_2, \cdots, y_n\}$，序列的斜率定义为

$$s_i=\frac{x_j-x_i}{\triangle t}\tag{4}$$

式中，x_i 和 x_j（$j>i$）是时间序列中的两个时间点的序列值，$\triangle t$ 表示两个时间点之间的间隔，值等于 $j-i$.

本文只考虑相邻时间的两点之间的序列斜率，则 $\triangle t=1$. 故其斜率为 $s_i=x_j-x_i$，$(j=i+1)$. 假设转化完后的序列为 $Z=\{z_1, z_2, \cdots, z_{n-1}\}$. 则转化过程为

$$z_i=\begin{cases}1 & x_{i+1}-x_i>0\\ 0 & x_{i+1}-x_i=0\\ -1 & x_{i+1}-x_i<0\end{cases}\tag{5}$$

设定当斜率大于 0 时，转化后序列值为 1；当斜率等于 0 时，转化后序列值为 0；当斜率小于 0 时，转化后序列值为 -1. 由上述转化之后，我们得到一个长度为 $n-1$ 的序列. 最终我们可以用 LCS 对转化之后的序列进行相似性度量. 例如，给定时间序列 $X=\{10, 30, 20, 20, 50, 30\}$. 则转化之后的序列为 $Z=\{1, -1, 0, 1, -1\}$. 如图 4 所示.

上述基于趋势信息的时间序列转换方法，虽然在其形式上十分简单，但可以解决时间序列数据分类中的若干问题.

（Ⅰ）在某些情况下有些同类的时间序列之间点对点函数距离较大，但序列变化趋势与形状相同；而不同类型的序列与其点对点函数距离较小，但形状以及变化趋势差异较大，导致的时间序列分类错误.

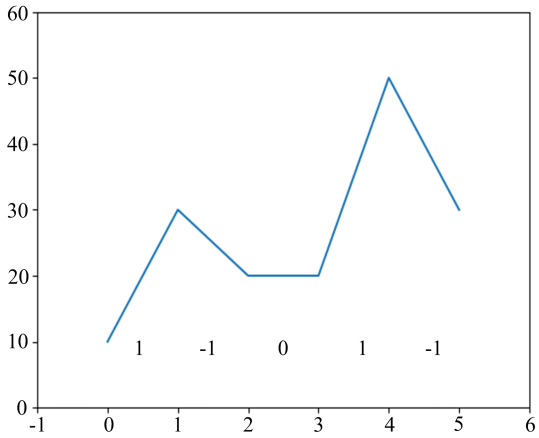

图 4 基于趋势的时间序列转换

Fig. 4 Trend-based time series conversion

如图 5 所示，我们从其形状和变化趋势看来，序列 X 和序列 Y 属于同一类，而序列 Z 则不属于该类. 但如果我们使用点对点的函数距离时，则会将序列 X 和 Z 划分为一类，而序列 Y 则在一定程度上被划分为其他类型. 如果利用趋势信息对原始序列进行离散化处理，可以很好地避免该种错误发生.

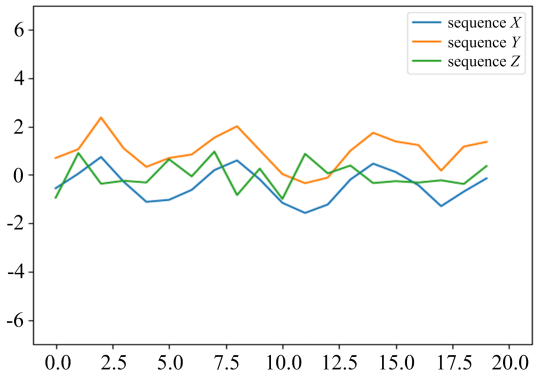

图 5 同类型序列数据变化趋势相同但是距离较远

Fig. 5 The same type of sequence has the same trends but farther away

当时间序列之间数值和变化趋势差异不大，但是变化序列波动幅度不同时，使用点对点函数距离或者使用序列的导数信息都不能很好地对原始时间序列进行分类.

如图 6 所示，序列 X 和序列 Y 属于同一类，但是两序列的波动幅度不同. 当用基于导数的时间序列分类算法时，由于两序列的导数值具有较大差异，所以很大概率会将两序列划分为不同的类型. 当我们利用趋势信息进行离散化处理之后，就不再受到原始序列值的影响. 由于只考虑序列的上升或者下降趋势，不考虑其震动幅度，因此在这种情况下，本文方法能更加有效地度量时间序列间的相似性.

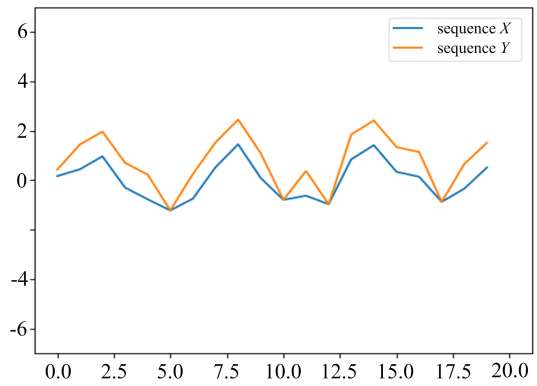

图 6 同类型序列波动幅度不同

Fig. 6 The same type of sequence fluctuation amplitude is different

基于趋势信息离散化的时间序列最长公共子序列算法有效地解决上述问题，能够对序列的变化趋势以及形状有很好的预测效果，但是由于离散化之后的序列不能够保留原始序列值的信息，使得其在某些数据集上效果较差. 故将本文提出将基于趋势信息的最长公共子序列算法与基于点对点函数距离的序列相似度度量方法相结合. 例如，DTW、MSM (move-split-merge)和 TWE 等.

2.2 基于趋势信息的 LCS 结合 DTW

由于基于趋势信息的 LCS 由于其本身不能保留原始时间序列值的信息，不能度量点对点之间的函数距离，导致在某些数据集上效果较差，为了提高分类准确率以及使其具有普遍适用性，我们将其与现有的点对点函数距离度量公式相结合，形成新的时间序列相似性度量公式，其具体表达形式为

$$D_{\mathrm{DTLCS}}(A,B)=\alpha D_{\mathrm{DTW}}(A,B)+\beta D_{\mathrm{TLCS}}(A,B) \tag{6}$$

式中，$D_{\mathrm{TLCS}}(A,B)$ 表示基于趋势信息的 LCS 相似度，且 α 与 β 满足 $\alpha+\beta=1$. 将 DTW 距离与基于趋势信息的 LCS 进行参数结合(DTLCS)，形成新的距离度量公式. 上述公式不仅能够保留原始的点对点函数距离度量信息，还能够有效地把握序列的趋势信息. 由于 $\alpha+\beta=1$，只需要考虑其中一个参数即可. 下面我们对参数 α 进行讨论.

当 $\alpha=0$ 时，上述公式则退化为基于趋势信息的 LCS；当 $\alpha=1$ 时，则退化为 DTW；当 $0<\alpha<1$ 时，参数 α 表示 DTW 距离所占的比重，即在进行时间序列分类预测时，两序列的点对点函数距离所占的比重. 由于点对点之间的函数距离往往较大，而我们所求出的序列趋势衡量距离是由 LCS 得出，其值较

小，所以一般情况下 α 的取值较小，防止掩盖序列的趋势信息. 当遇到某些时间序列数据其趋势信息有效性较小时，α 值将接近 1. 由公式(6)可以看出，理论上我们可以将 DTW 距离替换成我们所需要其他任何时间序列相似性度量公式，具有很好的普遍适用性. 本文仅对点对点函数距离进行结合.

算法 2.1 展示了 TLCS 结合 DTW 进行时间序列间的相似性度量. 首先对原始序列进行离散化(第 2、4 行)，使用无参数的 LCS 求转化之后的序列相似性值，并且与用 DTW 对原始序列进行相似性值相结合形成新的相似性值(第 6 行).

算法 2.1　Distance(s_1, s_2)

Input: s_1, s_2 / * 分别表示长度为 m 和 n 的序列 * /

Output: s_1 和 s_2 之间的弹性距离

1　for each i←1 to m do / * 序列 s_1 中的每个数值 * /

2　　$z_1[i]= s_1[i+1] - s_1[i]$

3　for each j←1 to m do / * 序列 s_2 中的每个数值 * /

4　　$z_2[j]=s_2[j+1] - s_2[j]$

5　$D_{Tlcs}(s_1, s_2)=D_{LCS}(z_1, z_2)$ / * D_{LCS} is 表示 LCS 距离 * /

/ * D_{dtw} 表示使用 DTW 时的距离 * /

6　$Distance(s_1, s_2)=\alpha * D_{dtw}(s_1, s_2)+\beta * D_{Tlcs}(s_1, s_2)$ / * α 和 β 参数通过在训练集上进行交叉验证的到最优参数 * /

3　实验

本节将用大量的实验来验证所提出的算法对时间序列分类的有效性，使用的数据集均来源于 UCR，表 1 对数据集中所含的测试实例个数、训练实例个数、序列长度、序列类属性个数以及序列类型进行介绍. 我们在 42 个数据集上进行时间序列的分类实验. 时间序列长度范围为 80 到 657；训练集(测试集)实例的数目范围为 16(20)到 600(861)；类属性的个数范围为 2 到 37 个. 包括 5 种类型的时间序列数据.

3.1　参数分析及选择

我们将最长公共子序列(LCS)、基于趋势信息的最长公共子序列(TLCS)、动态时间规整(DTW)、时间规整编辑(TWE)、移动分割合并(MSM)以及分别和 TLCS 相结合等时间序列相似度衡量方法应用于 1NN，进行时间序列分类预测. 最近邻分类器是一个十分简单的分类器，其将未分类的时间序列分类为与其最相似的类型. 尽管其简单，但是许多实验结果表明，与其他复杂的分类器相比，它能获得更好的分类效果[15].

首先对基于趋势信息的 LCS 时间序列分类参数进行调整. 由上面对 LCS 的介绍可以看出，在 LCS 中存在参数 δ 和 ε. 不同的 ε 对时间序列数据分类有较大的影响. 由于转化之后的序列全部为−1、0 和 1，两个点之间不存在匹配值的误差范围，故参数 ε 设置为 0. 对于参数 δ，其理论上最大值可以取到序列的最大长度，但是当其值超过一定界限时，对分类器的分类准确率影响不大. 图 7 给出在数据集 Beef、BeetleFly 和 Car 上不同 δ 对分类准确率的影响.

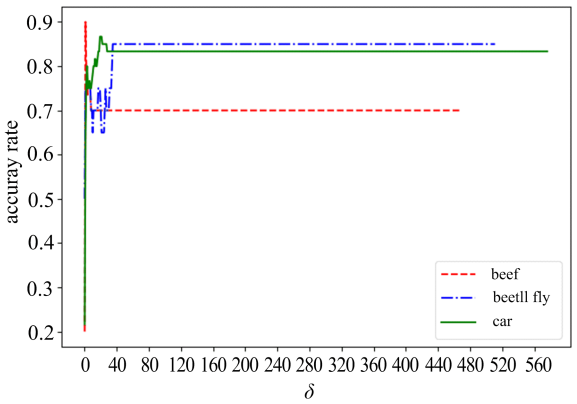

图 7　TLCS 中 δ 对分类准确率影响

Fig. 7　The effect of δ on classification accuracy

由图 7 可以看出，选择不同的 δ 时，数据集的分类准确率有波动. 当 δ 取值较小时，在 3 个数据集上，分类准确率曲线波动较大. 这是由于当 δ 取较小值时，表示序列相位偏移较小. 实际数据中，序列的偏移程度可能大于设置的 δ 阈值，导致分类准确率较低. 当 δ 较大时，分类准确率没有明显的波动. 因此，当我们对于一些未知的数据进行实验时，我们可以将其 δ 参数设为最大值，即为序列的长度，这样我们也能够获得较好的实验结果. 本节的实验选取 δ 为序列的长度，虽然该参数范围并不一定能得到最好的分类效果，但其与最优参数的分类准确率差距较小.

作为对照实验，我们用 LCS-1NN 对全部的 42 个数据集进行分类预测，如表 1 所示. 实验将 LCS 中的 δ 也设置为序列长度. 由于参数 ε 对分类准确率的影响较大，而且对于不同的数据集其最优的参数取值不同，因此需要对不同的数据集设置不同的参数. 对训练集进行 5 重交叉验证，选取平均准确率最高的参数作为其测试集上使用的参数.

表 1 数据集汇总

Tab. 1 Summary of dataset

DataSet	Train Size	Test Size	Length	No. of Class	Type
Adiac	390	391	176	37	IMAGE
ArrowHead	36	175	251	3	IMAGE
Beef	30	30	470	5	SPECTRO
BeetleFly	20	20	512	2	IMAGE
BirdChicken	20	20	512	2	IMAGE
Car	60	60	577	4	SENSOR
Coffee	28	28	286	2	SPECTRO
CricketX	390	390	300	12	MOTION
CricketY	390	390	300	12	MOTION
CricketZ	390	390	300	12	MOTION
DiatomSizeReduction	16	306	345	4	IMAGE
DistalPhalanxOutlineAgeGroup	400	139	80	3	IMAGE
DistalPhalanxOutlineCorrect	600	276	80	2	IMAGE
DistalPhalanxTW	400	139	80	6	IMAGE
ECG200	100	100	96	2	ECG
ECGFiveDays	23	861	136	2	ECG
FaceFour	24	88	350	4	IMAGE
Fish	175	175	463	7	IMAGE
GunPoint	50	150	150	2	MOTION
Ham	109	105	421	2	SPECTRO
Haptics	155	308	1 092	5	MOTION
Herring	64	64	512	2	IMAGE
InlineSkate	100	550	1 882	7	MOTION
ItalyPowerDemand	67	1 029	24	2	SENSOR
MiddlePhalanxOutlineAgeGroup	400	154	80	3	IMAGE
MiddlePhalanxOutlineCorrect	600	291	80	2	IMAGE
MiddlePhalanxTW	399	154	80	6	IMAGE
MoteStrain	20	1 252	84	2	SENSOR
OliveOil	30	30	570	4	SPECTRO
OSULeaf	200	242	427	6	IMAGE
Plane	105	105	144	7	SENSOR
ProximalPhalanxOutlineAgeGroup	400	205	80	3	IMAGE
ProximalPhalanxOutlineCorrect	600	291	80	2	IMAGE
ProximalPhalanxTW	400	205	80	6	IMAGE
SonyAIBORobotSurface1	20	601	70	2	SENSOR
SonyAIBORobotSurface2	27	953	65	2	SENSOR
SwedishLeaf	500	625	128	15	IMAGE
Symbols	25	995	398	6	IMAGE
ToeSegmentation1	40	228	277	2	MOTION
ToeSegmentation2	36	130	342	2	MOTION
Trace	100	100	275	4	SENSOR
TwoLeadECG	23	1 139	82	2	ECG

我们同样也用 DTW-1NN 对全部的 42 个数据集进行分类预测. 将 DTW 中的窗口设置为 0.1. 在此基础上,我们将 DTW 与本文提出的 TLCS 相结合(DTLCS),在数据集上进行实验. 该实验同样保持 DTW 的窗口值为 0.1,以便进行对比. 在这个实验中,参数 α 对时间序列分类准确率有较大的影响,在图 8 中我们保持 TLCS 中的 δ 值不变,给出了在不同参数 α 的情况下(其中 α 的步长为 0.1),DTLCS 在数据集 Beef、Car、BeetleFly 以及 Meat 上的分类准确率的变化曲线.

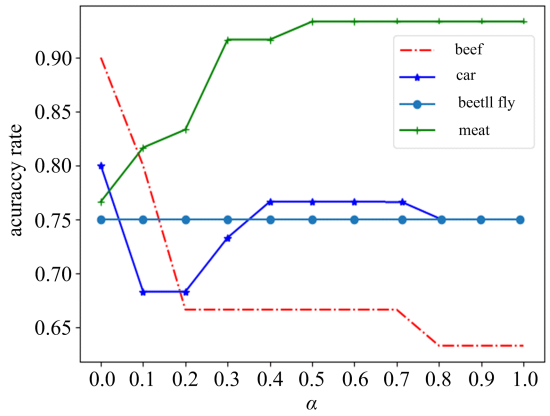

图 8　DTLCS 中 α 对准确率的影响

Fig. 8　The effect of α on DTLCS in accuracy

图 8 中,$\alpha=0$ 时,表示 TLCS 的分类准确率,$\alpha=1$ 时,表示 DTW 分类准确率. 从图 8 可看出,不同的时间序列数据对 α 的敏感程度不同;并且不同时间序列数据在进行分类预测时,其关注的重点有所不同. 对于 Beef 数据集,其分类准确率随着 α 值的增加而减少,而且存在较大的差异,这表明其侧重的是时间序列的变化趋势;与之相反,对于 Meat 数据集则侧重时间序列数据值的作用;而对于数据集 BeetleFly,不管 α 值如何变化,其最终分类准确率保持不变,这表明时间序列的变化趋势以及点对点函数距离均能很好地表示序列间的相似程度. 在 DTLCS 实验中,我们选取的 α 步长均为 0.1. 当在个别数据集上将 α 步长设置为 0.01 时,能有效地提高整体分类准确度,但其训练时间将大大增加. 为了提高训练速度,我们在进行 LTLCS(LCS 结合 TLCS)、MTLCS(MSM 结合 TLCS)和 TTLCS(TWE 结合 TLCS)实验时,初始设置的 α 步长为 0.1. 不过我们在个别数据集上将 α 的步长设置为 0.01,以便能得到更好的实验效果.

3.2　实验性能分析

表 2 给出了不同的时间序列相似度度量方案,使用最近邻分类器对数据集中的测试子集的分类准确率. TLCS 在 23 个数据集上比 LCS 表现较好,这表明单纯的衡量序列的变化趋势对时间序列数据也能有不错的分类效果;LTLCS 在 30 个数据集上优于 LCS,在 3 个数据集上没有差异;DTLCS 在 24 个数据集上比 DTW 表现较好,在 6 个数据集上没有差异;TTLCS 在 24 个数据集上比 TWE 表现较好,在 4 个数据集上没有差异;MTLCS 在 24 个数据集上比 MSM 表现较好,在 9 个数据集上没有差异. 我们所提出的 TLCS 与标准的 LCS 相比较其在某些数据集上有明显的提升,但在一些数据集上表现较差;但当我们将 TLCS 与 LCS、DTW、TWE 和 MSM 相结合时,在绝大多数的数据集上能有效提高其分类准确率.

表 2　测试准确率(%)Tab. 2　Testing accuracy rate(%)

DataSet	LCS	DTW	MSM	TWE	TLCS	LTLCS	DTLCS	MTLCS	TTLCS
Adiac	49.10	60.36	61.38	60.87	27.88	49.87(1.0)	59.34(0.4)	62.15(0.2)	61.13(0.2)
ArrowHead	77.71	70.29	72.57	80.57	79.43	**82.29(0.1)**	79.43(0.0)	79.43(0.0)	81.71(0.1)
Beef	26.67	63.33	53.33	66.67	**90.00**	**90.00(0.0)**	**90.00(0.0)**	**90.00(0.0)**	**90.00(0.0)**
BeetleFly	60.00	70.00	70.00	60.00	85.00	80.00(1.0)	**85.00(0.0)**	80.0(0.01)	65.00(0.01)
BirdChicken	55.00	75.00	75.00	65.00	**85.00**	**85.00(0.5)**	**85.00(0.0)**	80.00(0.1)	80.00(0.0)
Car	71.67	73.33	90.00	73.33	83.33	83.33(0.1)	68.33(0.1)	**91.67(0.04)**	73.33(0.4)
Coffee	96.43	**100.00**	92.86	**100.00**	96.48	92.86(0.2)	**100.00(0.1)**	92.86(0.1)	**100.00(0.1)**
CricketX	59.74	75.38	72.56	66.67	41.79	74.36(0.4)	**77.44(0.1)**	73.08(0.1)	66.67(0.1)
CricketY	61.03	74.36	72.05	62.31	48.21	**77.95(0.4)**	76.15(0.1)	72.56(0.2)	62.05(0.1)
CricketZ	56.15	75.38	74.87	65.90	47.69	75.13(0.6)	**77.44(0.1)**	73.33(0.2)	66.15(0.1)
DiatomSizeReduction	92.81	**96.73**	95.42	93.46	86.27	86.60(0.1)	95.42(0.1)	95.10(0.1)	93.79(0.1)
DistalPhalanxOutline AgeGroup	69.78	**76.98**	70.50	61.87	61.87	61.87(0.0)	73.38(0.2)	74.82(0.5)	64.03(0.1)

续表 2

DataSet	LCS	DTW	MSM	TWE	TLCS	LTLCS	DTLCS	MTLCS	TTLCS
DistalPhalanx OutlineCorrect	71.01	71.74	73.91	71.74	62.68	**75.00(0.5)**	70.65(0.6)	73.19(0.1)	72.10(0.1)
DistalPhalanxTW	60.43	58.98	63.31	61.87	53.96	60.43(1.0)	58.99(1.0)	**65.47(0.1)**	61.87(0.3)
ECG200	90.00	77.00	84.00	**91.00**	81.00	81.00(0.1)	81.00(0.0)	88.00(0.01)	**91.00(0.2)**
ECGFiveDays	76.07	76.77	80.14	86.76	85.48	84.32(0.1)	85.48(0.0)	85.48(0.0)	**87.22(0.01)**
FaceFour	**89.77**	82.95	87.50	84.09	80.68	**89.77(0.9)**	82.95(0.1)	86.36(0.1)	86.36(0.1)
Fish	87.43	82.29	**92.57**	78.29	86.86	89.14(0.3)	**90.86(0.0)**	92.57(0.01)	86.86(0.0)
GunPoint	94.67	90.67	96.67	91.33	94.67	95.33(0.3)	**98.67(0.1)**	**98.67(0.01)**	93.33(0.01)
Ham	54.29	46.67	58.10	57.14	**61.90**	60.95(0.2)	59.06(0.1)	59.06(0.0)	59.06(0.0)
Haptics	35.39	37.66	41.23	36.04	**45.78**	**45.78(0.0)**	**45.78(0.0)**	**45.78(0.0)**	**45.78(0.0)**
Herring	57.81	53.13	54.69	51.56	**60.94**	**60.94(0.0)**	59.38(0.1)	59.38(0.0)	59.38(0.0)
InlineSkate	41.64	38.36	42.73	34.55	**46.55**	45.27(0.3)	**46.55(0.0)**	42.36(0.1)	46.55(0.0)
ItalyPowerDemand	88.14	**95.04**	94.56	95.43	88.24	91.55(0.4)	**95.04(0.1)**	88.24(0.0)	95.72(0.2)
MiddlePhalanx OutlineAgeGroup	52.60	50.00	50.65	52.60	53.25	51.95(0.0)	53.25(0.5)	51.95(0.3)	**53.25(0.0)**
MiddlePhalanx OutlineCorrect	73.88	69.76	72.85	76.29	62.54	76.29(0.8)	70.45(0.7)	72.16(0.2)	**76.63(0.8)**
MiddlePhalanxTW	**58.44**	50.65	49.35	50.65	49.35	47.40(0.3)	49.35(0.0)	48.05(0.1)	50.65(0.9)
MoteStrain	81.23	83.47	87.62	84.82	90.73	**93.37(0.3)**	88.66(0.9)	90.73(0.0)	88.66(0.0)
OliveOil	46.67	83.33	**83.33**	86.67	50.00	43.33(0.0)	**83.33(0.2)**	**83.33(0.1)**	80.00(0.1)
OSULeaf	79.75	59.09	80.17	53.31	**80.58**	81.82(0.2)	80.58(0.0)	79.34(0.1)	**80.58(0.0)**
Plane	98.10	**100.00**	**100.00**	96.19	99.05	**100.00(0.6)**	**100.00(0.3)**	99.05(0.0)	97.14(0.0)
ProximalPhalanx OutlineAgeGroup	79.02	80.49	79.02	78.54	80.49	**84.88(0.1)**	79.51(0.2)	80.49(0.1)	78.54(0.1)
ProximalPhalanx OutlineCorrect	80.07	78.35	80.41	80.41	75.95	80.07(1.0)	78.69(0.4)	**81.44(0.1)**	79.04(0.2)
ProximalPhalanxTW	71.71	**76.10**	74.63	70.24	73.17	68.78(0.1)	73.17(0.0)	72.20(0.1)	69.76(0.7)
SonyAIBORobot Surface1	67.05	72.55	69.22	68.72	**76.87**	74.54(0.3)	63.33(0.1)	**76.87(0.0)**	69.55(0.1)
SonyAIBORobot Surface2	78.28	83.11	**87.09**	86.99	82.69	81.74(0.1)	82.69(0.0)	82.69(0.0)	82.69(0.0)
SwedishLeaf	85.60	79.20	87.04	79.20	77.76	87.68(0.3)	82.69(0.2)	**89.12(0.02)**	76.80(0.0)
Symbols	95.68	94.97	96.98	89.65	**98.19**	**98.19(0.0)**	**98.19(0.0)**	**98.19(0.0)**	**98.19(0.0)**
ToeSegmentation1	55.26	77.19	79.39	70.18	77.19	55.26(1.0)	77.63(0.1)	**81.58(0.1)**	69.74(0.1)
ToeSegmentation2	86.92	83.85	89.23	86.15	81.54	84.62(0.4)	87.69(0.1)	**92.31(0.1)**	86.15(0.1)
Trace	81.00	100.00	96.00	77.00	74.00	97.00(0.8)	**100.00(0.1)**	96.00(0.1)	77.0(0.1)
TwoLeadECG	76.82	90.42	96.75	74.71	73.84	94.82(0.8)	87.18(0.2)	**95.79(0.1)**	74.63(0.1)

表 3 是不同模型在所有数据集上的平均相对正确率. TLCS 相对于 LCS 在所有数据集的平均相对正确率为 2.02%，虽然 TLCS 提升效果不大，但其在分类时不需要训练参数，相较于 LCS 具有更强的抗干扰性. LTLCS 相对于 LCS 的平均相对准确率为 7.75%；DTLCS 相对于 DTW 平均相对正确率为 4.48%. MTLCS 相对于 MSM 平均相对准确率为 2.63%；TTLCS 相对于 TWE 相对准确率为 3.65%. 我们发现当将 TLCS 与 LCS 相结合时，能较好地提高其分类准确度. 这是因为两者都是使用最长公共子序列进行相似性度量，最终的相似性取值都为 0 到 1，能进行有效地结合；而其他方法最终

的相似性取值都是较大的数值，在进行结合时很容易将 TLCS 相似性值掩盖，我们可以改用更小的 α 步长来解决该问题. 为了更好地表现其结果，我们给出了其图形化比较方式，如图 9 所示. 由图 9 可以看出，TLCS 与 LCS 整体上分类准确度处于持平状态，但在个别数据集上明显优于 LCS；而 LTLCS 很明显要优于 LCS；此外 DTLCS、MTLCS、TTLCS 相较于 DTW、MSM 和 TWE 都有一定的提升，在个别数据集上提升效果较为明显.

表 3　平均相对准确率(%)

Tab. 3　Average relative accuracy(%)

	LCSS-TLCS / LCSS	LTLCS-LCSS / LCSS	DTLCS-DTW / DTW	MTLSC-MSM / MSM	TLCS-TWE / TWE
Mean	2. 02	7. 75	4. 48	2. 63	3. 65

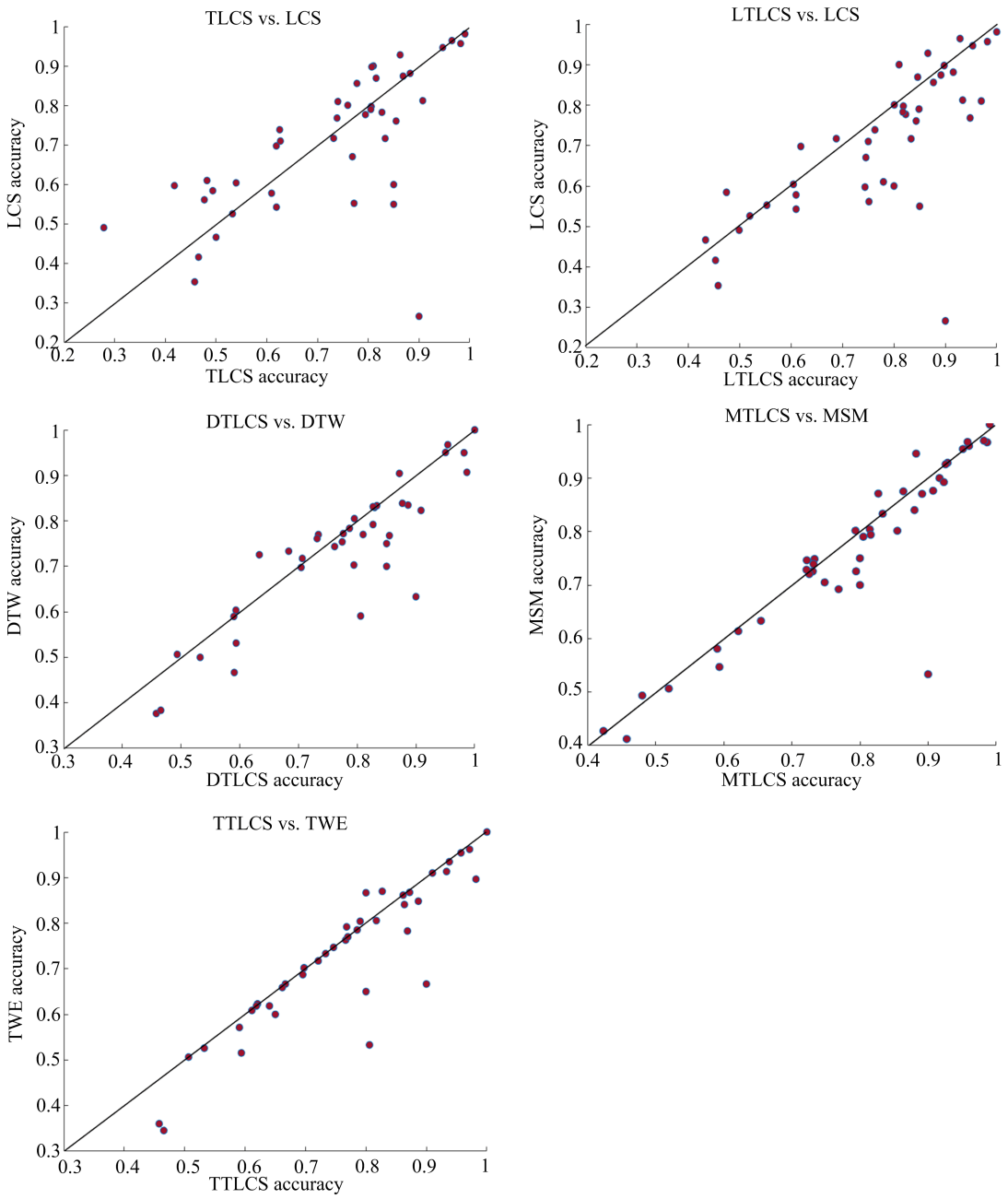

图 9　测试准确率比较

Fig. 9　Comparison of test accuracy

4 结论

本文提出了一种基于趋势信息的时间序列离散化方案,并用LCS对其进行相似性度量.相比于以往的基于点对点函数距离度量以及基于时间序列导数距离度量,该方法仅考虑序列的变化趋势,对现实生活中受各类噪声干扰的时间序列数据具有很强的抗干扰性.其缺点在于其不保留原始序列数据的数值信息.在大部分情况下,时间序列数据的数值信息也尤为重要,所以TLCS在某些数据集上效果较差.本文提出将TLCS与点对点距离度量方案相结合,有效地结合了序列的数值信息和变化趋势信息.实验结果表明,基于趋势信息的时间序列离散化表示方法能有效地表示时间序列的变化趋势.将其与已有的时间序列相似度度量方案相结合,能有效地提高时间序列分类准确率.

参考文献(References)

[1] 原继东.时间序列分类算法研究[D].北京:北京交通大学,2016.

[2] FALOUTSOS C, RANGANATHAN M, MANOLOPOULOS Y. Fast subsequence matching in time-series databases[J]. ACM SIGMOD Record, 1994, 23(2):419-429.

[3] RATANAMAHATANA C A, KEOGH E. Three myths about dynamic time warping data mining[C]// Proceedings of the SIAM International Conference on Data Mining. Newport Beach, USA: SIAM, 2005: 506-510.

[4] LATECKI L J, MEGALOOIKONOMOU V, WANG Q, et al. Partial elastic matching of time series[C]// Proceedings of the 5th International Conference on Data Mining. Houston, USA: IEEE, 2005: 701-704.

[5] JEONG Y S, JEONG M K, OMITAOMU O A. Weighted dynamic time warping for time series classification[J]. Pattern Recognition, 2011, 44(9): 2231-2241.

[6] KATE R J. Using dynamic time warping distances as features for improved time series classification[J]. Data Mining & Knowledge Discovery, 2016, 30(2): 283-312.

[7] SONG C, GE T. Window-chained longest common subsequence: Common event matching in sequences [C]// International Conference on Data Engineering. Seoul, South Korea: IEEE, 2015: 759-770.

[8] MARTEAU P F. Time warp edit distance with stiffness adjustment for time series matching[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2009, 31(2): 306-318.

[9] KEOGH E J, PAZZANI M J. Derivative dynamic time warping[C]// Proceedings of the First International Conference on Data Mining. Chicago: SIAM, 2001: 1-11.

[10] GÓRECKI T. Using derivatives in a longest common subsequence dissimilarity measure for time series classification[J]. Pattern Recognition Letters, 2014, 45(1): 99-105.

[11] SCHÄFER P. The BOSS is concerned with time series classification in the presence of noise[J]. Data Mining & Knowledge Discovery, 2015, 29(6): 1505-1530.

[12] 张建业,潘泉,张鹏.基于斜率表示的时间序列相似性度量方法[J].模式识别与人工智能, 2007, 20(2): 271-274.
ZHANG Jianye, PAN Quan, ZHANG Peng. Similarity measuring method in time series based on slope[J]. Pattern Recognition &Artificial Intelligence, 2007, 20(2): 271-274.

[13] RATANAMAHATANA C A, LIN J, GUNOPULOS D, et al. Mining time series data[J]. Data Mining & Knowledge Discovery Handbook, 2009:1069-1103.

[14] BAGNALL A, LINES J, BOSTROM A, et al. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances [J]. Data Mining & Knowledge Discovery, 2017, 31(3):6 06-660.

[15] 原继东,王志海,孙艳歌,等.面向复杂时间序列的K近邻分类器[J].软件学报, 2017, 28(11): 3002-3017.
YUAN Jidong, WANG Zhihai, SUN Yange, et al. K-nearest neighbor classifier for complex time series[J]. Journal of Software, 2017, 28(11): 3002-3017.