基于自适应增量学习的时间序列模糊聚类算法

王 玲[1,2],徐培培[1,2]
(1. 北京科技大学自动化学院,北京 100083; 2. 北京科技大学自动化学院
工业过程知识自动化教育部重点实验室,北京 100083)

摘 要: 针对现存可用于时间序列的增量式模糊聚类算法往往需要设置多个控制参数的问题,本文提出了一种基于自适应增量学习的时间序列模糊聚类算法. 该算法首先继承上一次聚类得到的簇结构信息以初始化当前聚类进程,然后在无需设置参数的情况下自适应地搜索当前数据块中的离群样本,并自动从离群样本创建新簇,最后检查空簇识别标识确定是否需要移除部分簇以保证后续聚类过程的效率. 实验结果表明所提算法对等长和不等长时间序列均具有良好的聚类准确性及运行效率.

关键词: 时间序列; 模糊聚类; 自适应增量学习; 离群样本
中图分类号: TP273 **文献标识码:** A **文章编号:** 0372-2112 (2019)05-0983-09
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2019.05.002

Adaptive Incremental Learning Based Fuzzy Clustering of Time Series

WANG Ling[1,2], XU Pei-pei[1,2]
(1. *School of Automation and Electrical Engineering, University of Science and Technology Beijing, Beijing* 100083, *China*;
2. *Key Laboratory of Knowledge Automation for Industrial Processes of Ministry of Education, School of Automation and Electrical Engineering, University of Science and Technology Beijing, Beijing* 100083, *China*)

Abstract: Existing incremental fuzzy clustering algorithms which can be used for time series often require setting multiple control parameters. To solve this problem, a fuzzy clustering algorithm of time series based on adaptive incremental learning is proposed. First, the cluster structure information obtained by the previous clustering process is inherited to initialize the current clustering process. Then, the outliers in current data block are adaptively searched without parameters, and new clusters are automatically created from the outliers. Finally, an empty cluster flag is checked to determine if some clusters need to be removed to ensure the favorable efficiency of subsequent clustering. The experimental results show that the proposed algorithm has good clustering accuracy and efficiency for both equal-length and unequal-length time series.

Key words: time series; fuzzy clustering; adaptive incremental learning; outliers

1 引言

时间序列是聚类问题中常用的数据类型之一,鉴于其高维特性,每条序列都能被视为一个包含大量数据点的复杂样本,对这种复杂样本进行聚类分析能够发现其中蕴含的有趣的模式信息[1]. 时间序列的聚类研究主要集中在序列的相似性度量以及聚类方法的选择上. 目前常用的欧式距离[2]、动态时间弯曲(Dynamic Time Warping, DTW)距离[3]、泊松相关系数[4]、最长公共子序列[5]等均可以根据具体情况应用于原始时间序列或映射到特征空间的降维表达序列上. 根据序列特点选择合适的相似性度量方法,再结合分层聚类[6]、划分聚类[7]、基于模型的聚类[8]、基于密度的聚类[9]等常见算法,就可以构造出对时间序列可用的聚类方法.

模糊聚类作为一种广泛使用的聚类技术,使样本以不同的强度隶属于多个簇,更符合实际,因此同样在

收稿日期:2018-07-19;修回日期:2018-11-11;责任编辑:蓝红杰
基金项目:国家自然科学基金(No. 61572073);北京科技大学中央高校基本科研业务费专项资金资助(No. FRF-BD-17-002A);北京市重点学科共建项目(No. XK100080537)

时间序列数据领域取得了众多研究进展[10~12]. 然而,大多数研究关注的都只是静态数据集,无法处理持续更新的动态数据,而基于增量学习的模糊聚类算法能够利用之前数据已取得的聚类结果对新增数据进行分批次地聚类,从而避免重复聚类带来的问题. 目前针对时间序列设计的增量式模糊聚类算法还很少,但存在一些应用于其他类型数据的研究可以扩展到等长时间序列的聚类中,例如文献[13]和[14]分别利用传统的和改进后的 FCM 算法对文本数据和网络日志数据进行增量聚类;文献[15]运用差分进化和随机森林实现了类别数据的增量聚类;文献[16]提出了一种新的模糊聚类目标函数,并结合数据块划分实现了对图像数据和恶意软件数据的增量聚类. 上述这些算法均需要数据类别的先验信息以提前设置簇个数,并且整个聚类过程中簇个数不再变化. 为解决这一问题,文献[17]提出了一种增量式 FCM(Incremental Fuzzy C-Means,IFCM)算法,对离线数据运用 FCM 得到初始聚类中心,然后对增量数据块中新搜集的数据进行模糊聚类. 文献[18]提出一种基于模糊聚类的动态数据挖掘(Dynamic Data Mining Based on Fuzzy Clustering,DDMFC)算法,以增量的形式处理新到达的数据,运行过程中不仅能够更新已有簇结构,还可以根据当前增量数据创建新簇并移除某些空簇. 以上基于增量学习的模糊聚类算法对等长的时间序列数据同样适用,但对于不等长时间序列却无能为力. 文献[19]中考虑到时间序列的不等长性,利用最长公共子序列对时间序列做相似性度量,借助分层聚类从离线数据中得到初始聚类个数,然后对增量数据进行模糊聚类,但存在需要设置参数才能识别离群样本的问题. 为了使算法在处理具体问题时具有更好的鲁棒性,一些文献中采用了自适应[25]的思想. 文献[26]基于新的模糊邻域函数,提出了一种自适应寻找具有任意形状、密度、分布和数量的聚类方法. 文献[27]提出了一种基于共享最近邻的局部自适应多核聚类方法. 这些方法均能够根据当前数据自身的特征自动调整处理方法或参数,从而获得最佳的处理效果.

基于上述自适应算法的思想,本文提出了一种基于自适应增量学习的时间序列模糊聚类算法(Adaptive incremental learning based fuzzy clustering of time series,AIFC). 对离线数据引入模糊聚类有效性评价指标来自动获取最佳初始簇个数,在此基础上,AIFC 算法能够根据增量数据块中的时间序列动态地更新现有簇结构、自动识别离群样本并增加新簇、移除部分空簇. 在聚类的增量学习过程中,除空簇移除阈值外,簇创建过程本身不需要设置任何参数,AIFC 算法自适应地结合之前聚类获取的簇结构信息来分辨离群样本及控制新簇的创建. 实验结果表明 AIFC 算法对等长和不等长时间序列都能得到良好的类别划分.

2 时间序列模糊聚类

模糊聚类是模糊集理论与聚类分析相结合的产物,常见的算法包括模糊 C 均值(FCM)、模糊 C 中心点、传递闭包、最大树等,其中 FCM 算法通过迭代地更新簇中心及隶属度矩阵实现目标函数的最小化来决定样本的归属,简洁、高效且应用最为广泛. FCM 算法的目标函数为:

$$J(\boldsymbol{U},V)=\sum_{i=1}^{N}\sum_{j=1}^{C}u_{ij}^{w}(dis(\boldsymbol{z}_i,\boldsymbol{v}_j))^2 \tag{1}$$

其中 $\boldsymbol{U}$ 是隶属度矩阵;V 是簇中心的集合;N 和 C 分别是样本和簇的个数;u_{ij} 是第 i 个样本 $\boldsymbol{z}_i$ 对第 j 个簇的隶属度;w 是模糊加权指数,一般取值为 2;$dis(\boldsymbol{z}_i,\boldsymbol{v}_j)$ 表示第 i 个样本 $\boldsymbol{z}_i$ 和第 j 个簇中心 $\boldsymbol{v}_j$ 之间的距离.

传统的 FCM 算法假定所有样本具有相同的维度,采用欧式(Euclidean,EU)距离作为距离度量函数,即 $dis(\boldsymbol{z}_i,\boldsymbol{v}_j)=dis_{EU}(\boldsymbol{z}_i,\boldsymbol{v}_j)=\|\boldsymbol{z}_i-\boldsymbol{v}_j\|_2$. 在满足 $u_{ij}\in[0,1]$,$\sum_{j=1}^{C}u_{ij}=1$ 的条件下,最小化目标函数 $J(\boldsymbol{U},V)$ 可得到簇中心和隶属度的计算公式:

$$\boldsymbol{v}_j=\frac{\sum_{i=1}^{N}u_{ij}^{w}\boldsymbol{z}_i}{\sum_{i=1}^{N}u_{ij}^{w}},j=1,2,\cdots,C \tag{2}$$

$$\begin{cases}\text{当 } dis(\boldsymbol{z}_i,\boldsymbol{v}_j)\neq 0 \text{ 时:}\\ u_{ij}=\left(\sum_{k=1}^{C}\left(\frac{dis(\boldsymbol{z}_i,\boldsymbol{v}_j)}{dis(\boldsymbol{z}_i,\boldsymbol{v}_k)}\right)^{\frac{2}{w-1}}\right)^{-1},i=1,2,\cdots,N\\ \text{当 } dis(\boldsymbol{z}_i,\boldsymbol{v}_j)=0 \text{ 时:}\\ u_{ij}=1,u_{ik}=0,k=1,2,\cdots,C \text{ 且 } k\neq j\end{cases} \tag{3}$$

然而,当被聚类样本是时间序列时,聚类分析问题根据应用领域的不同主要可分为整体聚类和子序列聚类两种,前者旨在根据相似性获得一系列相互独立的时间序列样本之间的类别划分,而后者是将一条长时间序列分段后得到的子序列视为样本,聚类的目的往往是寻找长时间序列中存在的模式或对子序列赋予类标签后得到长时间序列的降维符号化表达. 两种应用针对的时间序列样本均有可能是不等长的. 考虑到欧式距离无法度量不等长时间序列之间的相似性,引入 DTW 作为距离度量可以很好的解决这一问题[11]. DTW 距离根据时间序列的形态信息评估其相似性,这一点和欧式距离相同. 但是,与后者相比,前者能够通过拉伸或压缩某些序列片段得到两条时间序列之间的最优匹配,且不要求时间序列是等长的. 假设样本 $\boldsymbol{z}_i=\{z_{i1},z_{i2},\cdots,z_{iT_i}\}$ 是一条长度为 T_i 的时间序列,$\boldsymbol{v}_j=\{v_{j1},v_{j2},\cdots,v_{jT_j}\}$ 是长度为 T_j 的簇中心(也是时间序列),$\boldsymbol{z}_i$ 和 $\boldsymbol{v}_j$ 之

间的 DTW 距离需要通过逐步迭代来计算. 用两个维度为 $T_i \times T_j$ 的矩阵分别存储 $\boldsymbol{z}_i$ 和 $\boldsymbol{v}_j$ 中元素之间的欧式距离 $d(p,q) = |z_{ip} - v_{jq}|$ 和累计距离 $\gamma(p,q) = d(p,q) + \min\{\gamma(p-1,q),\gamma(p,q-1),\gamma(p-1,q-1)\}$,其中 $p = 1,2,\cdots,T_i$ 且 $q = 1,2,\cdots,T_j$. 迭代的计算累计距离即可得到两序列之间的 DTW 距离 $dis_{\mathrm{DTW}} = \gamma(T_i,T_j)$.

另外,当考虑 DTW 作为 FCM 算法的距离度量函数时,式(2)计算簇中心的方法就不再适用了. 针对这一问题,文献[11]中引入全局平均策略 DBA 计算一系列时间序列的均值,利用加权平均将 DBA 策略扩展到模糊聚类,能够在 DTW 距离的基础上精确而有效地得到时间序列簇的中心,对于等长或不等长时间序列均适用. 首先计算簇中心 $\boldsymbol{v}_j$ 和每条时间序列 $\boldsymbol{z}_i(i = 1,2,\cdots,N)$之间的 DTW 距离,图 1 所示为一个 DTW 路径的示例,$\alpha_{jq}(i)$表示矩阵上 $\boldsymbol{z}_i$ 和 $\boldsymbol{v}_j$ 之间 DTW 路径经过的纵坐标为 q 的单元个数,$Sum_{jq}(i)$表示这 $\alpha_{jq}(i)$个单元横坐标对应 $\boldsymbol{z}_i$ 中 $\alpha_{jq}(i)$个元素的和,则簇中心 $\boldsymbol{v}_j$ 的第 q 个元素 v_{jq}在模糊聚类过程中用如下公式进行更新:

$$v_{jq} = \frac{\sum_{i=1}^{N} u_{ij}^{w} Sum_{jq}(i)}{\sum_{i=1}^{N} \alpha_{jq}(i) u_{ij}^{w}} \tag{4}$$

对 $q = 1,2,\cdots,T_j$ 分别计算式(4),即可得到簇中心 $\boldsymbol{v}_j$.

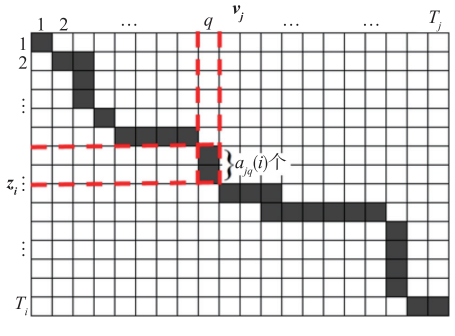


图1　簇中心和时间序列的DTW路径示例

3　基于自适应增量学习的时间序列模糊聚类算法(AIFC)

3.1　相关定义

本文基于传统的模糊聚类算法 FCM 进行扩展,结合增量学习思想对一些相关术语作出如下定义.

定义 1　增量数据块:对于数据总量很大且动态增长的时间序列数据集,将最先搜集到的一部分历史数据视为离线数据,记为 $DB[0]$,用于提供初始类别划分及其他相关信息. 除此之外,用数据块在线搜集动态到来的新数据,称为增量数据块,用 $DB[1]$,$DB[2]$,…表示,用于依次进行增量学习. 数据块的大小根据动态数据到来的速率来设定,可以设定为天、星期、月、年等时间单位.

定义 2　空簇识别标识:假定第 $h(h = 1,2,\cdots)$个数据块 $DB[h]$中共有 N_h 条时间序列,增量学习之后得到了 C_h 个簇. 由于可能存在数据块中所有时间序列对某个簇的隶属度均较小的情况,按照最大隶属度原则,没有新的序列划归到该簇中,因此被识别为空簇. 定义标识 $E^{(j)}$表示第 $j(j = 1,\cdots,C_h)$个簇在不断到来的数据块中被连续识别为空簇的次数,则 $E^{(j)}$计算为:

$$E^{(j)} = \begin{cases} E^{(j)} + 1, & \text{第 } j \text{ 个簇是空簇} \\ 0, & \text{其他} \end{cases} \tag{5}$$

这个标识将在增量学习过程中决定是否要把空簇移除以提高后续聚类过程的效率. 设定一个正整数阈值 ε,若某次增量学习后得到 $E^{(j)} \geqslant \varepsilon$,则把第 j 个簇移除.

定义 3　最大成员距离:根据最大隶属度原则将数据块 $DB[h]$中的每条时间序列 $\boldsymbol{z}_i(i = 1,2,\cdots,N_h)$划归到相应的簇后,用 $count^{(j)}$表示第 j 个簇中包含的序列条数,对于 $count^{(j)} \geqslant 2$ 的簇(排除了空簇和只包含一条序列的簇),定义其最大成员距离为:

$$D^{(j)} = \max\{dis(\boldsymbol{z}_g^{(j)},\boldsymbol{z}_l^{(j)})\},$$
$$\text{s.t.}\quad g,l = 1,\cdots,count^{(j)},g \neq l,count^{(j)} \geqslant 2 \tag{6}$$

其中 $\boldsymbol{z}_g^{(j)}$ 和 $\boldsymbol{z}_l^{(j)}$ 是第 j 个簇中任意两条不同的序列.

定义 4　合并距离:根据式(6)可计算出每个簇的最大成员距离,将其中的最大值称为合并距离,则合并距离计算为:

$$D_{h\_\max} = \max_{j=1,\cdots,C_h}\{D^{(j)}\} \tag{7}$$

定义 5　模糊半径:文献[20]中用簇中心与簇内样本之间的最大距离定义了簇半径,这里引入隶属度项构造模糊聚类所得簇的模糊半径. 对于数据块 $DB[h]$聚类后得到的 C_h 个簇,第 $j(j = 1,\cdots,C_h)$个簇的模糊半径计算为:

$$r_h^{(j)} = \max_{i=1,\cdots,N_h} u_{ij} dis(\boldsymbol{z}_i,\boldsymbol{v}_j) \tag{8}$$

其中 u_{ij}是时间序列 $\boldsymbol{z}_i$ 对第 j 个簇的隶属度,$dis(\boldsymbol{z}_i,\boldsymbol{v}_j)$是 $\boldsymbol{z}_i$ 和簇中心 $\boldsymbol{v}_j$ 之间的距离.

3.2　离线数据的聚类

离线数据为聚类的增量学习过程提供初始的簇结构信息,包括初始簇中心、模糊半径、合并距离. 本文对离线数据中的时间序列采用 FCM 算法进行模糊聚类,根据时间序列是否等长,分别选择欧式距离或 DTW 距离作为目标函数中的距离度量. 时间序列的每种形态都可以称为一种模式,而 FCM 算法存在需要提前设置好簇个数的问题,对于时间序列来说就是需要模式种类的先验信息. 为解决这一限制,这里结合 Kwon 指标[21]来获取最佳簇个数. Kwon 指标是一种结合了数据集几何特性的模糊聚类有效性评价指标,假定离线数据中包含 N_0 条时间序列,则 *Kwon* 指标计算为:

$$V_K = \frac{\sum_{i=1}^{N_0}\sum_{j=1}^{C} u_{ij}^{w} dis(\boldsymbol{z}_i,\boldsymbol{v}_j)^2 + \frac{1}{C}\sum_{j=1}^{C} dis(\boldsymbol{v}_j,\bar{\boldsymbol{v}})^2}{\min\limits_{j\neq k} dis(\boldsymbol{v}_j,\boldsymbol{v}_k)^2} \quad (9)$$

其中,$\bar{\boldsymbol{v}}$ 是 C 个簇中心的均值. 分子中的第一项评估的是类内紧凑度,第二项是避免簇个数 $C\rightarrow N_0$ 时指标值单调递减的惩罚函数;分母代表了类间的分离度. 对于簇个数 $C=2,3,\cdots,N_0$,分别对离线数据中的序列做 FCM 聚类并计算出相应的 V_K 值,V_K 值最小时对应的簇个数就是离线数据的聚类簇个数,记为 C_0.

令空簇识别标识 $E^{(j)}=0(j=1,2,\cdots,C_0)$,根据 V_K 值最小时对应的聚类结果计算合并距离 $D_0\_\max$ 及每个簇的模糊半径 $r_0^{(j)}(j=1,2,\cdots,C_0)$,结合 C_0 个簇中心,作为增量学习过程的初始信息.

3.3 AIFC 算法实现步骤

为了自适应地实现时间序列的增量式模糊聚类,本文提出的 AIFC 算法在结合离线数据聚类信息的基础上对增量数据块进行聚类,且每次聚类结束后获取的新信息都会被继承下来参与下一次增量数据块的聚类. AIFC 算法与其他算法相比的优势在于,簇的个数在整个聚类过程中是动态变化的,除空簇移除阈值外,簇创建过程本身不需要设置任何参数,完全依靠继承的簇结构信息来确定是否需要增加新簇. AIFC 算法的具体过程如算法 1 所示.

算法 1 AIFC

输入:增量数据块 $DB[h](h=1,2,\cdots)$,上一次聚类得到的 C_{h-1} 个簇中心 $\boldsymbol{v}_j(j=1,2,\cdots,C_{h-1})$、模糊半径 $r_{h-1}^{(j)}$、空簇识别标识 $E^{(j)}$,合并距离 $D_{h-1}\_\max$.

输出:$DB[h]$ 中时间序列的聚类结果.

1. 初始化簇个数 $C_h=C_{h-1}$,模糊半径 $r_h^{(j)}=r_{h-1}^{(j)}(j=1,2,\cdots,C_h)$;
2. 计算 $DB[h]$ 中的时间序列 $\boldsymbol{z}_i$ 和现有簇中心 $\boldsymbol{v}_j$ 之间的距离 $dis(\boldsymbol{z}_i,\boldsymbol{v}_j)$ 以及对现有簇的隶属度 u_{ij},得到加权距离 $weighted\_dis_{ij}=u_{ij}dis(\boldsymbol{z}_i,\boldsymbol{v}_j)$;
3. 计算簇中心之间的距离 $dis(\boldsymbol{v}_j,\boldsymbol{v}_k)(j,k=1,2,\cdots,C_h;j\neq k)$;
4. 对于 $j=1,2,\cdots,C_h$,若存在时间序列 $\boldsymbol{z}_i$ 同时满足如下两式:

$$weighted\_dis_{ij} > r_h^{(j)}$$

$$dis(\boldsymbol{z}_i,\boldsymbol{v}_j) > \frac{1}{2}\min_{\substack{k=1,\cdots,C_h\\ k\neq j}} dis(\boldsymbol{v}_j,\boldsymbol{v}_k)$$

则把 $\boldsymbol{z}_i$ 识别为远离现有簇的离群序列,执行 5. 否则,认为 $DB[h]$ 中不存在离群序列,执行 6.

5. 若只有一条离群序列,则创建一个新簇并将此序列视为新的簇中心,否则,为每个离群序列创建新簇,逐步将距离最近且小于 $D_{h-1}\_\max$ 的两个簇合并,分别计算簇中心,之后更新现有簇个数 C_h;
6. 对于 C_h 个簇中心,用 FCM 算法对 $DB[h]$ 进行迭代聚类,确定每条时间序列的归属并更新 $E^{(j)}$,若 $E^{(j)}>\varepsilon$,则移除第 j 个空簇,更新簇个数 C_h.
7. 计算每个簇的模糊半径 $r_h^{(j)}(j=1,2,\cdots,C_h)$ 以及合并距离 $D_h\_\max$.
8. 存储当前的 $\boldsymbol{v}_j(j=1,2,\cdots,C_h)$、$r_h^{(j)}$、$E^{(j)}$、$D_h\_\max$,输出聚类结果.

需要说明的是,步骤 4 中第一个公式用于判断时间序列 $\boldsymbol{z}_i$ 和所有当前簇中心之间的加权距离是否均大于该簇的模糊半径,第二个公式用于判断 $\boldsymbol{z}_i$ 和所有当前簇中心之间的距离是否均大于该中心与其他中心之间距离最小值的一半. 通过这两个条件,即可识别出当前增量数据块中的离群序列.

3.4 计算复杂度分析

AIFC 算法对每个增量数据块 $DB[h](h=1,2,\cdots)$ 中的时间序列进行聚类都可以分成 5 个部分. 首先考虑对等长时间序列采用欧式距离度量相似性,假定时间序列的长度为 T,C_{h-1} 表示数据块中的初始簇个数,C_h 表示增加新簇后总的簇个数,则计算距离矩阵和隶属度矩阵并得到加权距离所需的计算量为 $O(C_{h-1}\times N_h\times(T+1))$. 计算加权距离和簇中心之间的距离所需的计算量为 $O(C_{h-1}\times N_h\times(T+1)+{C_{h-1}}^2\times T)$. 利用加权距离、模糊半径及簇中心之间的距离识别数据块中的离群序列的计算量为 $O({C_{h-1}}^2+C_{h-1}\times N_h)$. 将找到的离群序列条数记为 $O_h(0\leqslant O_h<N_h)$,则根据离群序列创建新簇所需的计算量为 $O((O_h-C_h+C_{h-1})\times O_h^2\times(T+1))$. 用 FCM 算法对 C_h 个簇中心和 N_h 条时间序列进行迭代聚类,记迭代次数为 I,可得到整个迭代过程的计算量为 $O(I\times C_h\times N_h\times(T+2))$. 增量学习的最后需要更新所有簇结构信息及空簇识别标识以备下次聚类使用,需要的计算量为 $O(C_h\times(N_h+3))$. 综上所述,可以看出算法整体的计算复杂度与簇个数、离群序列条数、时间序列的长度及条数均密切相关,且在未识别到离群序列的情况下,算法总的计算量经简化后为 $O(((T+2)(I+1)+1)C_h\times N_h+(T+1)C_h^2)$.

另一方面,计算两个等长序列之间欧式距离的复杂度为 $O(T)$,基于欧式距离计算一个簇的簇中心的复杂度为 $O(N_h)$. 若数据块中的时间序列是不等长的,假定平均长度为 T,则在采用 DTW 作为距离度量的情况下,计算两序列之间距离的复杂度为 $O(T^2)$,计算簇中心的复杂度为 $O(N_h\times(T^2+T))$. 除此之外,两种距离度量方法对算法其他方面的计算复杂度均无影响.

4 实验结果

本节首先用 8 个时间序列分类数据集[22]对比测试了 AIFC 算法与其他两种增量模糊聚类算法对等长时间序列的聚类准确性及运行效率,然后又采用了一个真实的动态时间序列数据集[23],将 AIFC 算法应用到一元动态时间序列的符号化中,以验证对不等长时间序列的聚类效果及增量学习过程中簇个数的演变情况.

实验中所用算法涉及到 FCM 的部分,均设置模糊加权指数为 2,最大迭代次数为 50,迭代停止条件 $\max_{ij}\{|u_{ij}(l+1)-u_{ij}(l)|\}<0.01$($l$ 表示迭代次数). 所有实验均在 2.40GHz 处理器和 4.0GB 内存下运行的 Python3.5 环境中执行.

4.1　评价指标

(1)聚类精度 Acc:用于统计算法对已知类标签的样本类别划分正确的个数,公式如下:

$$Acc=\frac{\sum_{j=1}^{C}s_j}{N} \tag{10}$$

其中,s_j 表示在原数据集的第 j 个类别中,算法的聚类结果和实际数据分类结果相一致的样本个数,C 是原数据集中包含的类别个数,N 是数据集中的样本个数. Acc 值越大,算法聚类精度越高.

(2)Xie-Beni 指标:一种专门用于评价模糊聚类有效性的标准,用各样本到簇中心的距离之和来度量类内紧致性,用所有簇中心之间距离的最小值来度量类间分离性. 公式如下:

$$V_{XB}=\frac{\sum_{i=1}^{N}\sum_{j=1}^{C'}u_{ij}^{w}(dis(\boldsymbol{z}_i,\boldsymbol{v}_j))^2}{N\min\limits_{j\neq k}(dis(\boldsymbol{v}_j,\boldsymbol{v}_k))^2} \tag{11}$$

其中,C'是聚类算法得到的簇个数. 好的聚类划分应该使类内数据散布尽可能小,而使类间数据的隔离尽可能大,因此 V_{XB}值取极小值时对应最佳聚类.

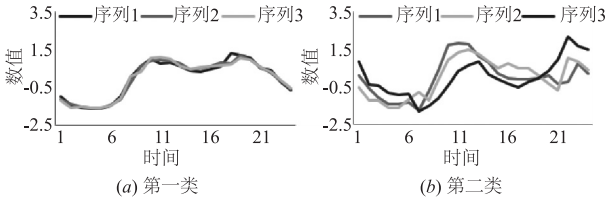
4.2　聚类效果测试

本节对比 AIFC 算法与其他两个增量模糊聚类算法 DDMFC[18] 和 IFCM[17] 对多个时间序列数据集的聚类效果. 实验中采用 UCR 数据库[22] 中的 8 个时间序列分类数据集,每个数据集都由训练集和测试集组成,细节如表 1 所示. 为了直观地展示数据集中时间序列的不同形态,图 2 和图 3 分别以 IPD 和 SC 数据集为例,从每个类别中随机选出 3 条时间序列用不同颜色曲线画出来,横轴“时间”表示序列中的每个时间点,纵轴“数值”表示序列在相应时间点处的值. 实验中将训练集和测试集分别视为离线、在线数据,并将测试集数据量的 10% 作为增量数据块,因此每个数据集都对应 10 个增量数据块. 由于每个数据集中的时间序列都是等长的,因此距离度量采用欧式距离.

AIFC 算法只有一个用于移除空簇的阈值需要设置. 由于移除空簇只是为了在数据总量未知时保证后续增量学习过程的效率,即被移除的簇只是不再参与后续聚类而不会被删除,而这里用的数据集都是数据总量已知的,为了更好的比较不同算法的聚类性能,统一将空簇移除阈值设置为 10,也就是说所有数据集的聚类过程中都不移除簇.

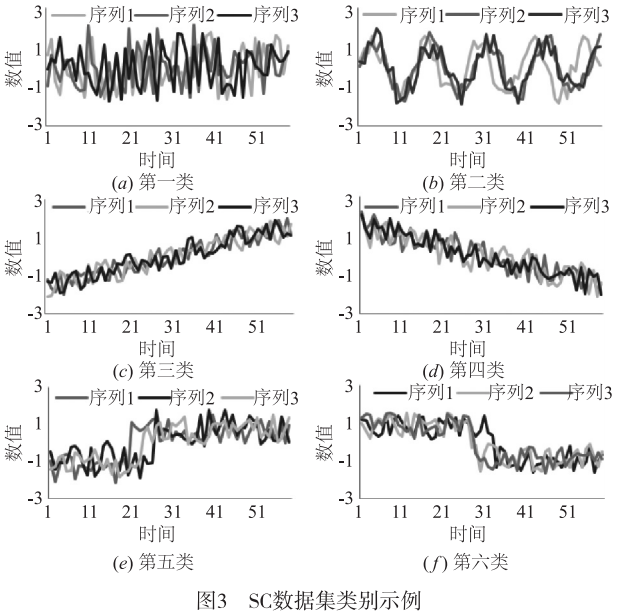
表 1　实验数据

数据集	类个数	时间序列长度	训练集大小	测试集大小
Italy Power Demand (IPD)	2	24	67	1029
Synthetic Control (SC)	6	60	300	300
Phalanges Outlines Correct (POC)	2	80	1800	858
CBF	3	128	30	900
ECG5000	5	140	500	4500
Chlorine Concentration (CC)	3	166	467	3840
Medical Images (MI)	10	99	381	760
Face All (FA)	14	131	560	1690



(a) 第一类　(b) 第二类

图2　IPD数据集类别示例



(a) 第一类　(b) 第二类　(c) 第三类　(d) 第四类　(e) 第五类　(f) 第六类

图3　SC数据集类别示例

DDMFC 算法需要设置 3 个参数,参数 α 借助隶属度和簇个数的倒数之间的差值作为判断数据是否离群的条件之一,参数 β 通过离群样本量和总样本量之间的比值控制是否要增加簇,参数 T 控制簇的移除. 实验中同样将参数 T 设置为 10,并采用文献[18]中实验部分的参数设置,设定 $\alpha=0.05$,$\beta=0.2$.

IFCM 算法需要设置两个参数,分别用于识别离群样本和控制新簇的创建. 由于这两个参数分别可以转换成 DDMFC 算法中的参数 α 和 β,因此这里对 IFCM 的实验同样采用参数设置 $\alpha=0.05$,$\beta=0.2$. 另外,IFCM 算法本身在增量学习过程中不会移除簇.

3 种增量模糊聚类算法对 8 个时间序列数据集的

聚类结果如表 2 所示,可以看出,AIFC 算法最终得到的簇个数是最接近原始数据集中类划分个数的,而且对每个数据集得到的聚类精度也是 3 种算法中最高的. 另外,由于 3 种增量模糊聚类算法都能够在增量学习过程中识别出少数离群样本,并根据离群样本来增加新的簇,所以在一些数据集中,算法最终得到的簇个数会比数据集实际的类划分个数要多.

表 2 算法对不同数据集的聚类个数及精度对比

数据集	AIFC		DDMFC		IFCM	
	簇个数	Acc	簇个数	Acc	簇个数	Acc
IPD	2	0.912	3	0.854	3	0.801
SC	6	0.941	6	0.901	6	0.883
POC	2	0.953	3	0.807	3	0.734
CBF	3	0.968	3	0.892	4	0.709
ECG5000	5	0.927	7	0.715	7	0.695
CC	4	0.890	6	0.606	6	0.601
MI	11	0.735	11	0.661	13	0.632
FA	14	0.812	16	0.733	15	0.750

为了对比各个算法的运行效率,图 4 展示了所有算法对各个测试集进行增量学习所消耗的时间,可以看出,AIFC 算法对 IPD、POC、CBF、ECG5000、CC、MI、FA 测试集的运行时间都是最少的,只在 SC 测试集中,3 种算法得到的簇个数一致,所以整体运行时间也近似相等. ECG5000、CC、FA 测试集本身包含的时间序列量较多且序列的长度比较长,MI 和 FA 测试集包含的簇个数较多,因此这 4 个测试集聚类消耗的时间远多于其他 4 个测试集. AIFC 算法在无需设置离群样本识别阈值和簇创建控制阈值的情况下并没有增加更多的时间消耗,且聚类结果也是最精确的,因此自适应性能最好,更具有优越性.

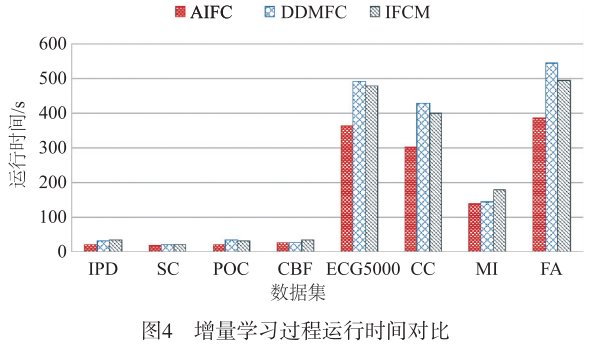


图4 增量学习过程运行时间对比

实验中将每个测试集划分成了 10 个增量数据块,为了更加直观的对比算法在增量学习过程中的聚类效果,图 5 以 IPD 数据集的测试集为例,展示了 3 种算法在每个数据块中聚类结果的 Xie-Beni 指标值,可以看出,AIFC 算法在每个数据块中得到的 V_{XB} 值都是最小的,表明与 DDMFC 算法和 IFCM 算法相比,AIFC 在每个数据块中的聚类划分最好,聚类效果更佳. DDMFC 算法在第 9 个数据块中创建了一个新簇,使得聚类中心个数由 2 变成了 3,因此得到的 V_{XB} 值相较于之前的数据块突然变大. IFCM 算法在第 6 个数据块中有新簇被创建,因此同样发生了 V_{XB} 值的突变.

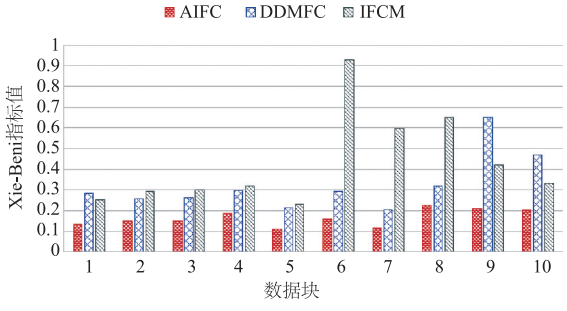


图5 增量学习过程聚类效果比较

4.3 簇个数演变测试

为了测试 AIFC 算法对不等长时间序列的聚类效果,以及加入空簇移除阈值后增量学习过程中簇个数的演变情况,本节将所提方法应用到一元动态时间序列的符号化中. 对一元动态时间序列进行分段聚类并用类标号表示每个子序列,可以实现符号化,且此方法可以扩展到多元,将数值型多元动态时间序列符号化为时态事务集.

实验采用记录了美国阿雷西沃地区水文气象数据的 Hydrometeorological 数据集[23],包含 WINDSPEED、DIR、GUSTS 3 个变量. 由于数据集中的数据至今仍在被记录,因此每个变量实际上都对应着一条一元动态时间序列. 实验中选取 2013 年 1 月 1 日至 3 月 31 日为期 3 个月的数据作为离线数据,记为 $DB[0]$,之后每个月获取的在线数据作为增量数据块 $DB[1]$,$DB[2]$,…. 文献[24]中提出了一种基于动态规划的分段方法,对多元时间序列数据具有良好的分段效果. 引入此分段算法对离线数据以及每个增量数据块中的多元时间序列分别进行分段,所得结果中,每个变量经分段后得到的子序列长度都不固定. 以 WINDSPEED 序列为例,采用 DTW 距离作为距离度量对离线数据中分段后得到的 280 条 WINDSPEED 子序列进行模糊聚类,Kwon 指标达到最小时得到最佳聚类划分个数为 4. 从每个簇中随机选择 3 条子序列用不同颜色的曲线展示在图 6 中,可以看出四个簇包含的序列形态具有明显差异.

然后,用 AIFC 算法依次对每个增量数据块中包含的 WINDSPEED 子序列进行自适应增量学习. 表 3 展示了空簇移除阈值设置为 2 时,离线数据集 $DB[0]$ 以及连续 15 个增量数据块 $DB[1]$,$DB[2]$,…,$DB[15]$ 在聚类过程中簇个数的演变情况,其中子序列条数是用分段算法对每个数据子集中的多元时间序列进行分段得到的,V_{XB} 值是根据每个数据子集的聚类结果计算出的.

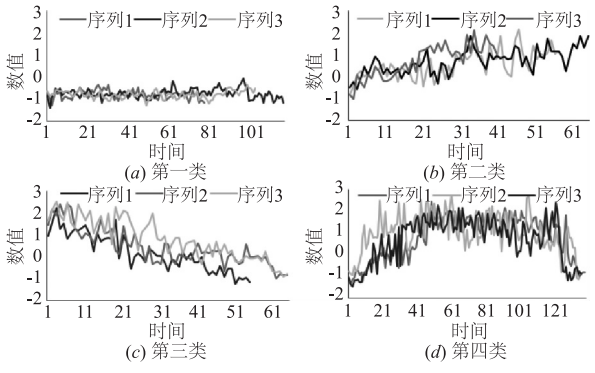


图6　离线数据中WINDSPEED序列的分段聚类结果示例

由表 3 可以看出,簇的个数在增量学习过程中是动态变化的,AIFC 算法只要找到离群序列就会新创建至少一个簇,且当某个簇在连续两个数据块中都被识别为空簇时,就会在下一数据块中被移除,不再参与后续聚类以保证算法的运行效率.

表 3　聚类过程中簇个数的演变

	子序列条数	簇个数	离群序列个数	新创建的簇	空簇	移除的簇	V_{XB} 值
DB[0]	280	4	–	–	–	–	0.341
DB[1]	96	4	0	0	0	0	0.312
DB[2]	95	5	3	1	0	0	0.607
DB[3]	95	6	2	1	0	0	0.841
DB[4]	101	6	0	0	1	0	0.732
DB[5]	93	8	5	2	1	0	1.105
DB[6]	89	7	0	0	0	1	0.884
DB[7]	97	7	0	0	1	0	0.697
DB[8]	93	8	4	1	2	0	1.019
DB[9]	95	8	2	1	0	1	0.983
DB[10]	99	8	0	0	0	0	0.801
DB[11]	91	9	4	1	2	0	1.199
DB[12]	95	9	0	0	2	0	0.845
DB[13]	100	8	2	1	0	2	0.828
DB[14]	98	8	0	0	0	0	0.702
DB[15]	96	9	1	1	2	0	1.003

为了更加直观的展现簇个数的演变对算法运行效率的影响,图 7 给出了 AIFC 算法对每个增量数据块中 WINDSPEED 子序列的运行时间,结合表 3 中的数据可知,在每个增量数据块中包含的子序列条数相差不大的情况下,聚类得到的簇个数越多,花费的时间就越长,而空簇移除阈值的设置能够及时移除部分空簇,使得参与后续聚类的簇个数不至于过多,从而保证了算法的运行时间不会一直增长,最终所提方法对每个增量数据块的运行时间稳定在 500s 左右.

被移除的簇只是不再参与后续聚类过程,在空簇移除阈值设置为 2 的情况下,表 3 所示聚类结果实际上一共产生了 13 个簇(*DB*[15]中的簇个数加上整个过程移除的簇个数). 为了评估空簇移除阈值取不同值对算法运行结果的影响,图 8 给出了不同空簇移除阈值设置下,AIFC 算法对 *DB*[1],*DB*[2],…,*DB*[15]聚类结果的平均 V_{XB} 值、总体运行时间、总体簇个数的变化情况. 当空簇移除阈值设置的比较小时,例如取 1,每个增量数据块中识别出的空簇都会被立刻移除,因此聚类过程的总体运行时间和平均 V_{XB} 值都比较小. 但这种情况下若下一增量数据块中含有该空簇代表的模式,就需要重新创建新簇,使得最终得到的总体簇个数偏多且其中含有部分重叠簇. 当赋予较大的空簇移除阈值时,部分空簇由于在之后的某一增量数据块中有新的子序列加入,因而不会被移除也不用创建新簇,使得重叠簇问题得到缓解,最终得到的总体簇个数就比较少. 但同时由于空簇的存在,整个聚类过程的运行时间就会增加,且平均 V_{XB} 值也会变大,表明每个增量数据块的聚类效果变差. 实际应用中可以根据算法的运行结果来权衡空簇移除阈值的设置,期望获得较好的运行效率和聚类效果时,应当将阈值设置的小一些,而期望结果

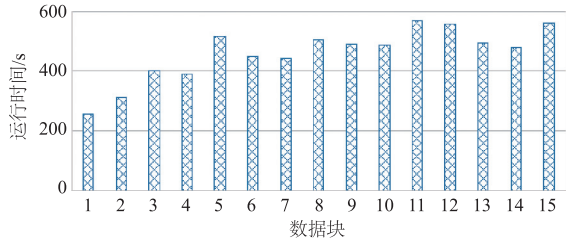


图7　增量学习过程消耗的时间

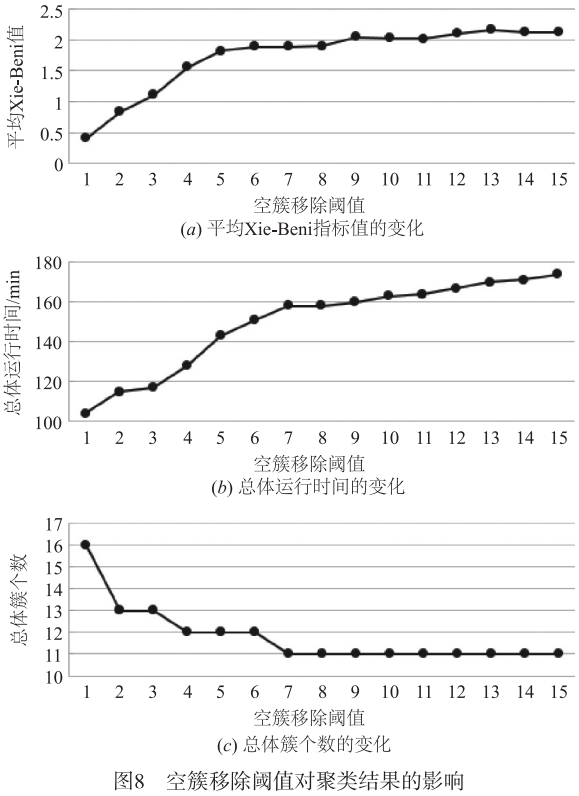


图8　空簇移除阈值对聚类结果的影响

中重叠簇较少时,可以适当增大阈值.

AIFC 算法对 WINDSPEED 子序列进行自适应增量学习后,分别给每个簇中包含的子序列赋予相同的类标号,即可增量的获取到一元动态时间序列的符号表达. 以数据块 *DB*[1]和 *DB*[3]为例,分别从中抽取连续 5 天的 WINDSPEED 数据记录,分段以及子序列符号化的结果如图 9、图 10 所示,其中字母表示子序列转换成的符号,括号中的数值是子序列对所属簇的隶属度. 结合表 3 中的簇个数演变可知,WINDSPEED 子序列在数据块 *DB*[1]和 *DB*[3]中分别被划分成了 4 和 6 类,因此图 10 中序列的符号化表达比图 9 中多了两个符号 E 和 F,代表新出现的两种序列形态. 可以看出,对于分段后得到的子时间序列,借用 AIFC 算法能够很好的将子序列按变化形态划分类别,进而得到整体序列的符号化表达.

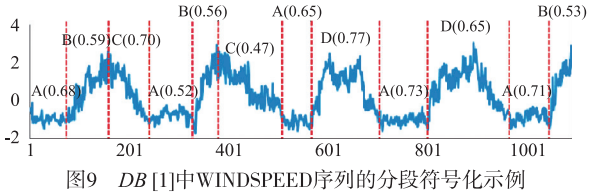


图9 *DB*[1]中WINDSPEED序列的分段符号化示例

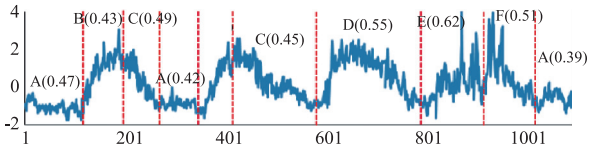


图10 *DB*[3]中WINDSPEED序列的分段符号化示例

5 结论

本文基于传统的 FCM 聚类算法进行扩展,提出了一种基于自适应增量学习的时间序列模糊聚类算法 AIFC. 考虑到 FCM 自身的缺陷,对离线数据引入 Kwon 指标确定最佳簇个数并得到初始簇结构信息作为增量学习过程的初始输入. AIFC 算法在继承了前一次聚类结果的基础上,根据当前增量数据块中的时间序列更新、创建、移除簇结构,且识别离群序列和创建新簇均无需人工干预即可自适应完成. 实验结果表明,为了捕捉时间序列之间的形态相似性,AIFC 算法对等长和不等长时间序列分别利用不同的距离度量,均取得了良好的聚类效果. 未来将研究利用 AIFC 算法把多元动态时间序列符号化为时态事务集,进而挖掘模糊时态关联规则以反映数据中隐藏的知识信息.

参考文献

[1] Aghabozorgi S, Shirkhorshidi A S, Wah T Y. Time-series clustering-A decade review [J]. Information Systems, 2015, 53: 16 – 38.

[2] Izakian H, Pedrycz W. Agreement-based fuzzy C-means for clustering data with blocks of features [J]. Neurocomputing, 2014, 127: 266 – 280.

[3] Bankó Z N, Abonyi J N. Correlation based dynamic time warping of multivariate time series [J]. Expert Systems with Applications, 2012, 39(17): 12814 – 12823.

[4] Tseng V S, Kao C P. Efficiently mining gene expression data via a novel parameterless clustering method [J]. IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB), 2005, 2(4): 355 – 365.

[5] Górecki T. Using derivatives in a longest common subsequence dissimilarity measure for time series classification [J]. Pattern Recognition Letters, 2014, 45: 99 – 105.

[6] Zhou F, De la Torre F, Hodgins J K. Hierarchical aligned cluster analysis for temporal clustering of human motion [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2013, 35(3): 582 – 596.

[7] Huang X, Ye Y, Xiong L, et al. Time seriesk-means: A new k-means type smooth subspace clustering for time series data [J]. Information Sciences, 2016, 367: 1 – 13.

[8] Cherif A, Cardot H, Boné R. SOM time series clustering and prediction with recurrent neural networks [J]. Neurocomputing, 2011, 74(11): 1936 – 1944.

[9] Shao J, Hahn K, Yang Q, et al. Combining time series similarity with density-based clustering to identify fiber bundles in the human brain [A]. Data Mining Workshops (ICDMW), 2010 IEEE International Conference on [C]. IEEE, 2010. 747 – 754.

[10] D Urso P, Maharaj E A. Autocorrelation-based fuzzy clustering of time series [J]. Fuzzy Sets and Systems, 2009, 160(24): 3565 – 3589.

[11] Izakian H, Pedrycz W, Jamal I. Fuzzy clustering of time series data using dynamic time warping distance [J]. Engineering Applications of Artificial Intelligence, 2015, 39: 235 – 244.

[12] Aghabozorgi S, Wah T Y, Amini A, et al. A new approach to present prototypes in clustering of time series [A]. The 7th International Conference of Data Mining [C]. Las Vegas, USA, 2011. 214 – 220.

[13] Aghabozorgi S R, Wah T Y. Using incremental fuzzy clustering to web usage mining [A]. International Conference of SOCPAR'09 [C]. IEEE, 2009. 653 – 658.

[14] Mei J P, Wang Y, Chen L, et al. Incrementalfuzzy clustering for document categorization [A]. 2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE) [C]. IEEE, 2014. 1518 – 1525.

[15] Saha I, Maulik U. Incremental learning based multiobjective fuzzy clustering for categorical data [J]. Information Sciences, 2014, 267: 35 – 57.

[16] Wang Y, Chen L, Mei J P. Incremental fuzzy clustering with multiple medoids for large data[J]. IEEE Transactionson Fuzzy Systems, 2014, 22(6): 1557 – 1568.
[17] Tudu B, Ghosh S, Bag A K, et al. Incremental FCM technique for black tea quality evaluation using an electronic nose[J]. Fuzzy Information and Engineering, 2015, 7(3): 275 – 289.
[18] Crespo F, Weber R. A methodology for dynamic data mining based on fuzzy clustering[J]. Fuzzy Sets and Systems, 2005, 150(2): 267 – 284.
[19] Aghabozorgi S, Saybani M R, Wah T Y. Incremental clustering of time-series by fuzzy clustering[J]. Journal of Information Science and Engineering, 2012, 28(4): 671 – 688.
[20] 王玲,孙华. 基于自适应学习的演化聚类算法[J]. 控制与决策, 2016(3): 423 – 428.
WANG Ling, SUN Hua. Evolving clustering method based on self-adaptive learning[J]. Control and Decision, 2016(3): 423 – 428. (in Chinese)
[21] Kwon S H. Cluster validity index for fuzzy clustering[J]. Electronics Letters, 1998, 34(22): 2176 – 2177.
[22] Chen Y, Keogh E, Hu B, et al. The UCR Time Series Classification Archive[EB/OL]. http://www.cs.ucr.edu/~eamonn/time_series_datd/, 2015 – 07.
[23] Dua D, Karra Taniskidou E. UCI Machine Learning Repository[EB/OL]. http://archive.ics.uci.edu/ml/index.php, 2017.
[24] Guo H, Liu X, Song L. Dynamic programming approach for segmentation of multivariate time series[J]. Stochastic Environmental Researchand Risk Assessment, 2015, 29(1): 265 – 273.
[25] Shan D, Xu X, Liang T, et al. Rank-adaptive non-negative matrix factorization[J]. Cognitive Computation, 2018, 10(3): 506 – 515.
[26] Du M, Ding S, Xue Y. A robust density peaks clustering algorithm using fuzzy neighborhood[J]. International Journal of Machine Learning and Cybernetics, 2018, 9(7): 1131 – 1140.
[27] Ding S, Xu X, Fan S, et al. Locally adaptive multiple kernel k-means algorithm based on shared nearest neighbors[J]. Soft Computing, 2017, 22(14): 4573 – 4583.

作者简介

王　玲　女,1974 年生于北京. 北京科技大学自动化学院副教授. 研究方向为数据挖掘、机器学习.
E-mail: lingwang@ustb.edu.cn

徐培培　女,1994 年生于安徽淮北. 硕士研究生,研究方向为数据挖掘、机器学习.