控制与决策
Control and Decision

面向密度分布不均数据的近邻优化密度峰值聚类算法

陈蔚昌, 赵嘉, 肖人彬, 王晖, 崔志华

引用本文:
陈蔚昌,赵嘉,肖人彬,王晖,崔志华. 面向密度分布不均数据的近邻优化密度峰值聚类算法[J]. 控制与决策, 2024, 39(3): 919–928.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2022.1151

面向密度分布不均数据的近邻优化密度峰值聚类算法

陈蔚昌[1], 赵 嘉[1†], 肖人彬[2], 王 晖[1], 崔志华[3]

(1. 南昌工程学院 信息工程学院，南昌 330099；2. 华中科技大学 人工智能与自动化学院，武汉 430074；3. 太原科技大学 计算机科学与技术学院，太原 030024)

摘 要: 密度分布不均数据是指类簇间样本分布疏密程度不同的数据. 密度峰值聚类(DPC)算法在处理密度分布不均数据时,倾向于在密度较高区域内找到类簇中心,并易将稀疏类簇的样本分配给密集类簇. 为避免上述缺陷,提出一种面向密度分布不均数据的近邻优化密度峰值聚类(DPC-NNO)算法. DPC-NNO算法结合逆近邻和k近邻定义新的局部密度,提高稀疏样本的局部密度,使算法能更准确地找到类簇中心;定义分配策略时引入共享近邻,计算样本间相似性,构造相似矩阵,使同一类簇样本联系更紧密,避免错误分配样本. 将所提出的DPC-NNO算法与IDPC-FA、DPCSA、FNDPC、FKNN-DPC、DPC算法进行对比,实验结果表明,DPC-NNO算法在处理密度分布不均数据时能获得优异的聚类效果,对于复杂数据集和UCI数据集,DPC-NNO算法的综合性能优于对比算法.

关键词: 密度峰值；聚类分析；密度分布不均；逆近邻；共享近邻；样本相似性

中图分类号: TP301.6 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2022.1151

引用格式: 陈蔚昌,赵嘉,肖人彬,等. 面向密度分布不均数据的近邻优化密度峰值聚类算法[J]. 控制与决策, 2024, 39(3): 919-928.

Density peaks clustering algorithm with nearest neighbor optimization for data with uneven density distribution

CHEN Wei-chang[1], *ZHAO Jia*[1†], *XIAO Ren-bin*[2], *WANG Hui*[1], *CUI Zhi-hua*[3]

(1. School of Information Engineering，Nanchang Institute of Technology，Nanchang 330099，China；2. School of Artificial Intelligence and Automation，Huazhong University of Science and Technology，Wuhan 430074，China；3. School of Computer Science and Technology，Taiyuan University of Science and Technology，Taiyuan 030024，China)

Abstract: Data with uneven density distribution are those where the distribution of samples varies in sparsity between class clusters. When dealing with uneven density datasets, the density peak clustering (DPC) algorithm tends to find the center of class clusters in the higher density area and assign samples from sparse class clusters to dense class clusters. To avoid these defects, this paper proposes a density peaks clustering algorithm with nearest neighbor optimization (DPC-NNO) for data with uneven density distribution. The DPC-NNO algorithm combines the reverse nearest neighbor and k-nearest neighbor to define a new local density that improves the local density of sparse samples, allowing the algorithm to find class cluster centers more accurately; shared nearest neighbors are introduced to define the assignment strategy to calculate the similarity between samples and construct a similarity matrix to make the samples of the same class clusters more closely related and avoid the wrong assignment of samples. In this paper, we compare the DPC-NNO algorithm with IDPC-FA, DPCSA, FNDPC, FKNN-DPC, and DPC algorithms. Experimental results show that the DPC-NNO algorithm can achieve excellent clustering results on uneven density datasets, and the comprehensive performance of the DPC-NNO algorithm is better than other comparison algorithms on complex datasets and UCI datasets.

Keywords: density peaks；clustering analysis；uneven density distribution；reverse nearest neighbor；share nearest neighbor；similarity of samples

收稿日期: 2022-07-01；**录用日期:** 2022-12-01.

基金项目: 国家自然科学基金项目 (52069014, 51669014)；科技创新——2030“新一代人工智能”重大项目 (2018AAA0101200).

责任编委: 刘宝碇.

†通讯作者. E-mail: zhaojia925@163.com.

0 引 言

聚类是数据挖掘中不可或缺的一类方法,它是根据相似性将样本分成多个类簇的过程,同一类簇样本间具有最大的相似性,不同类簇样本间的相似性较小. 聚类能够对数据进行分析,寻找出其中隐藏的信息,在日常生活工作中得到了广泛应用,如交通事故分析[1]、入侵检测[2]、图像分割[3-4]和电力检测[5]等.

随着聚类分析技术的发展,学者们根据不同原理提出了多种聚类算法[6],主要分为基于划分[7]、基于层次[8]、基于密度[9]、基于网格[10]等算法. 在基于划分的聚类算法中,k-means[11]和k-medoids[12]算法比较典型,两种算法原理简单易理解,对简单数据集效果较好,不同之处在于类簇中心的选取方式,前者将类簇内样本位置的平均值作为类簇中心,后者将距平均值最近的样本选定为类簇中心. 基于划分的聚类算法需提前确定类簇数目和类簇中心,并且受噪声点的影响较大. 在基于密度的聚类算法中,DBSCAN算法[13]将某密度区域作为中心,不断地向周围密度可达区域生长,最后形成最大密度相连的样本集合,该算法聚类高效且能够处理任意形状的样本,不足之处是聚类质量对参数的选取较敏感.

2014年,Rodriguez等[14]提出了密度峰值聚类(density peaks clustering. DPC)算法. DPC算法找到的类簇中心是密度峰值,周围是密度低的样本,类簇中心之间相距较远. 该算法的优点是能够快速发现任意形状数据集的类簇中心,并分配剩余样本,从而快速完成聚类过程. DPC算法的不足之处是: 1)在计算样本的局部密度时,截断距离内的其他样本对该样本的密度贡献很大,未综合考虑类簇样本分布情况,易找到错误的类簇中心; 2)在样本分配过程中,将剩余样本分配给密度比它高且距离它最近样本所在的类簇,若其中一个样本分配出错,则易产生连锁反应,导致大量样本分配错误.

近年来,针对DPC算法存在的缺陷,学者们进行了深入研究并提出多种改进算法[15]. 针对DPC算法的局部密度问题,Yuan等[16]提出了一种基于k近邻的自适应融合密度峰值聚类算法(KNN-ADPC). KNN-ADPC在定义局部密度时,使用k近邻更合理地划分样本,并且在聚类过程中自适应选取截断距离,提高了聚类质量. Cheng等[17]提出了基于最小生成树的局部密度峰值聚类(LDP-MST)算法. LDP-MST算法使用局部密度峰值构造最小生成树,反复切割最长的边,直至数量与类簇数相同,它能够很好地处理流形数据集. Tao等[18]提出了基于全局和局部一致性可调流形距离的密度峰值聚类(MDPC)算法. MDPC算法引入流形距离计算样本的局部密度,通过改变流形距离的指数项和标度因子,调整不同类簇样本间的距离之比,更容易找到样本类簇中心,从而获得不错的聚类结果. Cheng等[19]提出了基于局部核共享邻域的流形数据集密度峰值聚类改进(SLORE-DP)算法. SLORE-DP算法重新定义了样本与其共享邻居之间的距离,避免了计算数据集中所有样本间的距离,降低了算法的时间和空间复杂度.

针对DPC算法分配策略存在的问题,赵嘉等[20]提出了基于相互邻近度的密度峰值聚类(DPC-MND)算法. DPC-MND算法在设计分配策略时引入k近邻,计算样本间的相互邻近度,构建邻近度矩阵分配剩余样本,使样本分配更加准确. Yu等[21]提出了基于加权局部密度序列和最近邻分配的密度峰值聚类(DPCSA)算法. DPCSA算法在设计分配策略时使用边界条件,将样本间的欧氏距离与所有样本间的欧氏距离平均值作比较,将分配策略分为两个阶段以减少样本分配错误. 孙林等[22]提出了基于k近邻和优化分配策略的密度峰值聚类算法. 该算法引入高密度最近邻和密度差值建立相似度矩阵,且用邻域、相似集和相似域等概念,使样本的分配过程更加合理. Yu等[23]提出了基于证据理论的三向密度峰聚类方法(3W-DPET). 3W-DPET在分配剩余样本时,将剩余样本分为3个不相交的区域,分别为正区域、边界区域和负区域,利用证据理论的构造和收集k近邻信息,有效地为样本分配真实标签. 综上所述,DPC及其改进算法在进行数据聚类时,能取得不错的聚类效果,但其局部密度定义方式和分配策略仍有一定的缺陷. 另外,针对特定数据集的聚类效果不佳.

密度分布不均数据广泛存在于实际生活中,如异常数据分析、语音识别、图像分割等. 密度分布不均数据特点如下:样本分布复杂,组成的类簇疏密程度不同. DPC及其改进算法已广泛地应用于较多领域,但在处理密度分布不均数据时,DPC算法忽视了样本所处环境,未综合考虑类簇样本的分布情况. DPC算法用核函数计算稀疏类簇样本的局部密度都较小,导致选择的类簇中心大多都处于密集类簇,类簇中心的确定与局部密度紧密联系,因此,DPC算法难以正确找到密度分布不均数据的类簇中心;DPC算法在分配剩余样本时,稀疏类簇的样本易被视为异常点,或将剩余样本就近分配给密度较大的样本,将稀疏类簇的样本分配给密集类簇,导致聚类效果不佳.

为此,本文提出面向密度分布不均数据的近邻优

化密度峰值聚类(density peaks clustering with nearest neighbor optimization for data with uneven density distribution. DPC-NNO)算法. DPC-NNO算法在定义局部密度时引入逆近邻和k近邻,增强稀疏类簇样本间的联系,同时考虑全部样本间的相互影响,使算法更容易找到稀疏类簇的密度峰值;在设计分配策略时引入共享近邻,构造样本相似性矩阵,增强类簇样本间的联系,从而在分配剩余样本时更加准确.

1　密度峰值聚类算法

DPC算法能够快速发现任意形状数据集的密度峰值. 其特点是: 1)类簇中心的局部密度较大,围绕类簇中心的样本密度较小; 2)类簇中心距离较远,能避免相互影响. DPC算法计算样本的局部密度时有两种计算方法:针对大规模数据应使用如下截断核公式(1);针对小规模数据应使用如下高斯核公式(2):

$$\rho_i=\sum_j \chi(d_{ij}-d_c),\ \chi(x)=\begin{cases}0,\ x\geqslant 0;\\ 1,\ x<0.\end{cases} \tag{1}$$

$$\rho_i=\sum_{i\neq j}\exp\Big(-\frac{d_{ij}^2}{d_c^2}\Big). \tag{2}$$

其中: d_{ij}表示样本x_i与x_j之间的欧氏距离; d_c是截断距离,需人为设定,截断距离对样本局部密度的影响很大. 在截断核形式的局部密度中,样本x_i的局部密度与其截断距离范围内的样本个数有关;在高斯核形式的局部密度中,其他样本对样本x_i密度的影响随着距离的增加而削减.

DPC算法在分配剩余样本时,找到一个局部密度比该样本高且距它最近的样本,把这两个样本之间的距离称为相对距离δ_i,其定义为

$$\delta_i=\min_{j:\rho_j>\rho_i}(d_{ij}). \tag{3}$$

当该点为密度峰值时,相对距离δ_i定义为

$$\delta_i=\max_{i\neq j}(\delta_j). \tag{4}$$

根据式(1)～(4)计算出样本x_i的局部密度ρ_i和相对距离δ_i,将ρ_i设为横轴、δ_i为纵轴,建立决策图,在图中选取ρ_i和δ_i值均较大的样本为类簇中心. 但是,很多时候仅通过观察决策图很难正确选取密度峰值,这时可将决策值γ_i大的点作为密度峰值. γ_i的计算公式为

$$\gamma_i=\rho_i\cdot\delta_i. \tag{5}$$

找到类簇中心后,将剩余样本分配给密度比它大且距离它最近样本所在的类簇,从而实现了对样本的高效分配.

2　DPC-NNO算法

本节介绍DPC-NNO算法的主要思想,包括逆近邻和k近邻的局部密度,共享近邻样本相似性的分配策略. 本文使用的符号及其含义如表1所示.

表1　符号及其含义

符号	含义
d_{ij}	样本x_i与x_j之间的欧氏距离
d_c	截断距离
ρ_i	样本x_i的局部密度
δ_i	样本x_i的相对密度
γ_i	样本x_i的决策值
$\mathrm{KNN}(x_j)$	样本x_i的k近邻集合
$\mathrm{RNN}(x_i)$	样本x_i的逆近邻集合
$Z(x_i,x_j)$	样本x_i与x_j的隶属度
$\mathrm{SNN}(x_i,x_j)$	样本x_i与x_j的共享近邻集合
$\omega(x_i,x_j)$	样本x_i与x_j的邻近度
$A(x_i,x_j)$	样本x_i与x_j的相互邻近度
$\mathrm{Sim}(x_i)$	样本x_i与x_j的相似度

2.1　逆近邻和k近邻的局部密度

定义1　逆近邻(reverse nearest neighbor, RNN). 在数据集X中,若样本$x_i,x_j\in X$且样本x_i在样本x_j的k近邻集合$\mathrm{KNN}(x_j)$中,则样本x_j为样本x_i的逆近邻. 表示为

$$\mathrm{RNN}(x_i)=\{x_j\in X|x_i\in\mathrm{KNN}(x_j)\}. \tag{6}$$

定义2　样本间的隶属度$Z(x_i,x_j)$. 对于样本x_i和x_j的隶属度$Z(x_i,x_j)$定义为

$$Z(x_i,x_j)=\begin{cases}\exp(-(d_{ij})^2),\ x_j\in\mathrm{KNN}(x_i);\\ \dfrac{\exp(-d_{ij})}{\exp(-d_{ij})+1},\ x_j\notin\mathrm{KNN}(x_i).\end{cases} \tag{7}$$

定义3　逆近邻和k近邻的局部密度. 对于样本x_i的局部密度定义为

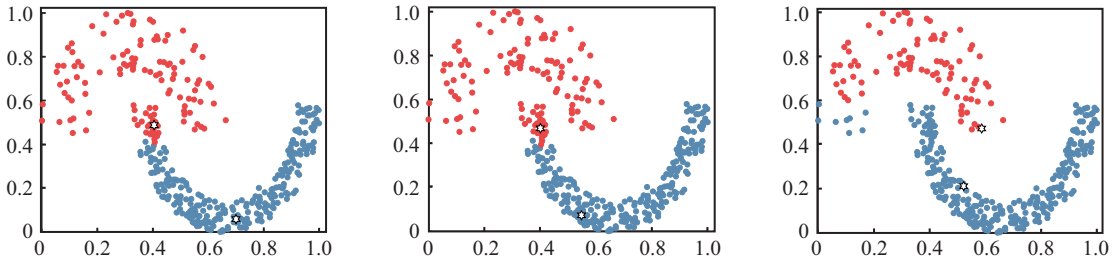
$$\rho_i=|\mathrm{RNN}(x_i)|+\frac{1}{k}\cdot\Big[\sum_{x_j\in\mathrm{KNN}(x_i)}Z(x_i,x_j)+\sum_{x_j\notin\mathrm{KNN}(x_i)}Z(x_i,x_j)\Big]. \tag{8}$$

其中: $|\mathrm{RNN}(x_i)|$为样本x_i的逆近邻个数,第2部分为x_i与k近邻样本x_j之间的隶属度求和,第3部分为x_i与非k近邻的样本x_j之间的隶属度求和.

在用式(8)计算样本x_i的局部密度时,第1部分考虑逆近邻对样本x_i的贡献,若该点的逆近邻个数越多,则说明该点被越多的样本包围,从而该点的局部密度越大,计算样本x_i的局部密度时该部分权重

较大. 第2部分计算x_i的k近邻点对局部密度的贡献, 表示k个近邻点的离群程度之和, 该值越大, 则该样本的局部密度越大. 第3部分计算非k近邻点对样本x_i局部密度的影响, 该部分对密度的影响较小. 本文定义的局部密度在计算样本x_i的密度时, 引入逆近邻、均衡近邻点和非近邻点对样本x_i的影响, 综合考虑所有样本的相互影响, 使得稀疏类簇的样本具有较大的密度, 从而容易找到稀疏类簇的类簇中心.

为验证本文提出的逆近邻和k近邻的局部密度比高斯核、截断核的局部密度更容易找到密度分布不均数据的类簇中心, 对Jain数据集进行聚类, 该数据集的两个类簇疏密情况不同. 图1是不同局部密度定义方式在Jain数据集上找到的类簇中心, 其中用六角星表示类簇中心.



(a) 高斯核的局部密度 (b) 截断核的局部密度 (c) 逆近邻和k近邻的局部密度

图 1 不同局部密度定义方式在Jain数据集上找到的类簇中心

如图1(a)和图1(b)所示, DPC算法的高斯核的局部密度和截断核的局部密度均没在稀疏类簇找到类簇中心; 而本文提出的逆近邻和k近邻的局部密度, 综合考虑了样本的分布特点, 增强了稀疏类簇内样本的密度, 容易找到稀疏类簇中心, 如图1(c)所示. 由此可见, 使用本文所提出的局部密度能够找到稀疏类簇的类簇中心.

2.2 共享近邻样本相似性的分配策略

定义 4 共享近邻(share nearest neighbor, SNN). 在数据集X中, $x_i, x_j \in X$, 样本x_i和x_j的k近邻集合分别为$\mathrm{KNN}(x_i)$和$\mathrm{KNN}(x_j)$, $\mathrm{KNN}(x_i)$与$\mathrm{KNN}(x_j)$的交集称为x_i和x_j的共享近邻集合, 表示为

$$\mathrm{SNN}(x_i, x_j) = \mathrm{KNN}(x_i) \bigcap \mathrm{KNN}(x_j). \tag{9}$$

定义 5 样本间的相似度$\mathrm{Sim}(x_i, x_j)$, 其定义如下:

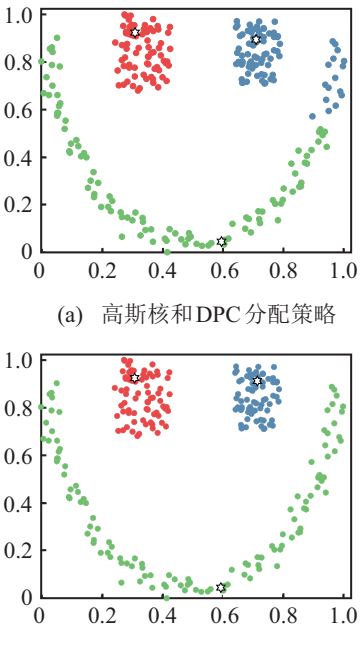
$$\omega(x_i, x_j) = \begin{cases} \mathrm{e}^{-d_{ij}^2}, \ j \in \mathrm{KNN}(i); \\ 0, \ \text{otherwise}. \end{cases} \tag{10}$$

$$A(x_i, x_j) = \frac{\sum\limits_{v \in [\mathrm{KNN}(i), i]} \omega_{vj} + \sum\limits_{v \in [\mathrm{KNN}(j), j]} \omega_{vi}}{k}. \tag{11}$$

$$\mathrm{Sim}(x_i, x_j) = |\mathrm{SNN}(x_i, x_j)| \cdot A(x_i, x_j). \tag{12}$$

其中: $\omega(x_i, x_j)$表示样本x_i与样本x_j的邻近度, 将样本x_i与其他样本之间的关系分为k近邻点和非近邻点两种情况, $\omega(x_i, x_j)$仅考虑两样本之间的欧氏距离; $A(x_i, x_j)$表示样本x_i与样本x_j的相互邻近度, 是样本x_i及其k近邻到x_j点的ω, 与样本x_j及其k近邻到x_i点的ω求和, 再归一化, 反映了样本所处环境的密集程度; $\mathrm{Sim}(x_i, x_j)$为样本x_i与样本x_j的相似度; $|\mathrm{SNN}(x_i, x_j)|$表示样本x_i与样本x_j的共享近邻个数. 在计算样本相似度时综合考虑了样本之间的分布特性, 其样本间共享近邻样本越多, 样本相似度越高, 使得算法能够正确分配剩余样本.

为验证所提出的分配策略对密度分布不均数据的有效性, 对Lineblobs数据集进行聚类. 图2(a)为使用高斯核和DPC的分配策略得到的聚类结果, 图2(b)为采用高斯核和共享近邻相似性的分配策略得到的聚类结果.



(a) 高斯核和DPC分配策略

(b) 高斯核和共享近邻相似性分配策略

图 2 高斯核与不同分配策略组合在Lineblobs数据集上的聚类结果

如图2(a)所示,Lineblobs数据集的两个方形类簇样本有较高的局部密度,根据DPC算法的分配策略,易将下方“U”型簇的部分样本错误地分配给密集程度较高的方形类簇. 本文提出的共享近邻样本相似性分配策略,使同一类簇样本间具有较高的相似性,增强了稀疏类簇样本之间的联系,如图2(b)所示,使用本文所提出的分配策略能正确分配“U”型稀疏类簇样本.

2.3　算法步骤

输入: 数据集X,样本的k近邻个数;

输出: 最终的聚类结果,给出样本的类簇标签.

step 1: 对数据进行预处理和归一化.

step 2: 计算样本间的欧氏距离,构造样本间的距离矩阵.

step 3: 根据式(3)和(4)计算样本x_i的相对距离δ_i,根据式(6)～(8)计算样本x_i的局部密度ρ_i.

step 4: 根据式(5)计算样本的决策值并画出决策图,选取类簇中心.

step 5: 根据式(9)～(12)计算样本相似度,构造共享近邻相似度矩阵.

step 6: 在相似度矩阵中找出与已分配样本相似度最大的未分配样本,将它们分配到已分配样本所在类簇.

step 7: 当所有已分配样本与未分配样本之间的相似度不为0时,转至step 6;否则转至step 8.

step 8: 当存在未分配的样本时,按照DPC算法的分配策略分配;否则转至step 9.

step 9: 聚类结束,输出最终结果.

2.4　算法复杂度分析

本节将分析算法的时间复杂度. 设数据集的样本规模为n,类簇数为m,样本近邻的数量设置为k.

DPC算法的时间复杂度主要由以下4部分组成: 1)计算样本间的距离矩阵; 2)计算每个样本的局部密度; 3)计算每个样本的相对距离; 4)分配样本. 前3部分的时间复杂度均为$O(n^2)$,第4部分的时间复杂度为$O(n)$,因此DPC算法的时间复杂度为$O(n^2)$.

DPC-NNO算法时间复杂度主要由以下6部分构成: 1)计算样本间欧氏距离的时间复杂度为$O(n^2)$,对欧氏距离进行升序排列的时间复杂度为$O(n\log_2 n)$; 2)计算样本相对距离需要的时间复杂度为$O(n^2)$; 3)寻找样本间共享近邻点以及是否为逆近邻点的复杂度均为$O(n^2)$; 4)各样本密度由该样本的逆近邻个数与样本间的隶属度求和,需要的时间复杂度为$O(n)$; 5)计算相似性矩阵的时间复杂度为$O(n^2)$,但当样本间无共享近邻时,样本间相似性为0,故仅需计算样本间共享区域存在样本的相似性,所以该部分实际时间复杂度要低于$O(n^2)$; 6)分配类簇中心的时间复杂度为$O(m)$,分配剩余样本需要的时间复杂度为$O(n^2)$. 综上,DPC-NNO算法与DPC算法的时间复杂度量级相同,均为$O(n^2)$.

3　实验结果与分析

3.1　实验设置

为验证DPC-NNO算法的有效性,本文采用8个密度分布不均数据集、10个复杂数据集以及8个UCI数据集进行实验. 将DPC-NNO算法与DPC[14]、IDPC-FA[24]、DPCSA[21]、FNDPC[25]、FKNN-DPC[26]算法进行比较. 除DPCSA和IDPC-FA算法无需对参数进行调优外,其他算法均在实验过程中调参后得到最佳结果. 其中: DPC-NNO和FKNN-DPC算法的近邻数k在1～50之间选取,步长为1; FNDPC算法的参数在0.01～1之间选取,步长为0.01; DPC算法的参数在0.1%～5%之间选取,步长为0.1%. 文中所有实验的硬件环境均为11th Gen Intel(R) Core(TM) i5-11 400 H @ 2.70 GHz 2.69 GHz、16 G RAM、Windows 11的64位操作系统,编程环境为Matlab R2018b.

本文对聚类效果的评价指标为基于调整互信息(adjusted mutual information, AMI)[27]、调整兰德系数(adjusted rand index, ARI)[27]和Fowlkes-Mallows指数(Fowlkes-Mallows index, FMI)[28]. 各项指标值最优值为1,越接近1,算法对数据集的聚类效果越好. 实验结果表中: “Arg-”表示算法处理数据集时参数的最优值,“–”表示该算法不需要进行调参. 加粗的指标值表示在处理同一数据集时该结果为最大值.

3.2　密度分布不均数据集的实验结果分析

本文实验的密度分布不均数据集共8个,它们的特点如表2所示. 表3为6种算法对上述数据集聚类结果的评价指标. 从表3可以看出: DPC-NNO算法在处理Jain、Cth3、Ring、Cmc、Ls、Lineblob数据集的评价指标均为1,聚类效果都是最佳; 处理Compound、

表2　密度分布不均数据集

数据集	样本规模	数据维数	类簇个数
Jain	373	2	2
Compound	399	2	6
Cth	1 016	2	3
Ring	1 000	2	2
Cmc	1 002	2	3
Ls	1 741	2	6
Pathbased	300	2	3

表3 6种算法在8个密度分布不均数据集上的聚类结果评价指标值

聚类算法	Jain				Compound			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NNO	**1**	**1**	**1**	11	**0.876 2**	**0.882 9**	**0.915 0**	9
IDPC-FA	**1**	**1**	**1**	–	0.792 2	0.832 7	0.881 5	–
DPCSA	0.216 7	0.044 2	0.592 4	–	0.839 2	0.828 4	0.870 7	–
FNDPC	0.596 1	0.725 7	0.905 1	0.47	0.851 6	0.868 4	0.904 6	0.38
FKNN-DPC	0.709 2	0.822 4	0.935 9	43	0.846 7	0.843	0.888 4	7
DPC	0.618 3	0.714 6	0.881 9	0.8	0.825 0	0.636 2	0.723 6	4.6

聚类算法	Cth3				Ring			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NNO	**1**	**1**	**1**	4	**1**	**1**	**1**	5
IDPC-FA	0.875 8	0.832 7	0.878 6	–	**1**	**1**	**1**	–
DPCSA	0.789 1	0.653 8	0.754 7	–	**1**	**1**	**1**	–
FNDPC	0.875 8	0.832 7	0.878 6	0.45	0.550 8	0.565 1	0.789 2	0.47
FKNN-DPC	**1**	**1**	**1**	22	**1**	**1**	**1**	6
DPC	0.686 6	0.513 5	0.647 3	0.1	0.351 7	0.301 9	0.685 7	0.1

聚类算法	Cmc				Ls			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NNO	**1**	**1**	**1**	12	**1**	**1**	**1**	5
IDPC-FA	0.809 3	0.842 1	0.902 7	–	0.707 6	0.627 4	0.732 5	–
DPCSA	0.665 6	0.576 1	0.745 4	–	0.725 2	0.599 9	0.712 9	–
FNDPC	0.809 3	0.842 1	0.902 7	0.28	0.756 4	0.689 8	0.780 8	0.37
FKNN-DPC	**1**	**1**	**1**	49	0.871 9	0.817 9	0.873 5	48
DPC	0.385 7	0.266 1	0.537 7	5	0.766 5	0.689 4	0.777 9	0.9

聚类算法	Pathbased				LineBlobs			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NNO	**0.948 2**	**0.969 5**	**0.979 6**	5	**1**	**1**	**1**	5
IDPC-FA	0.844 2	0.859 3	0.906 7	–	**1**	**1**	**1**	–
DPCSA	0.707 3	0.613 3	0.751 1	–	**1**	**1**	**1**	–
FNDPC	0.575 1	0.506 7	0.706 5	0.01	0.779 4	0.717 9	0.814 8	0.11
FKNN-DPC	0.930 5	0.949 9	0.966 5	9	**1**	**1**	**1**	7
DPC	0.557 3	0.508 2	0.684 8	0.4	0.837 5	0.823 7	0.884 2	4.2

Pathbased数据集时,虽然评价指标值没有达到1,但效果比其他对比算法好. 综上所述,说明DPC-NNO算法对密度分布不均数据集的聚类效果好.

图3为DPC-NNO算法和5种对比算法对密度分布不均Cmc数据集的聚类结果. 图3中用六角星表示类簇中心,不同类簇用不同颜色进行区分. Cmc数据集由一个较密集椭圆类簇和两个稀疏圆弧类簇组成. 对于处理Cmc数据集: DPC-NNO和FKNN-DPC算法能得到正确的聚类结果; IDPC-FA和FNDPC算法虽能找到正确的类簇中心,但在分配剩余样本时出错; 而DPCSA和DPC算法均在椭圆类簇中找到两个类簇中心,导致聚类结果极差.

3.3 复杂数据集的实验结果分析

为进一步验证DPC-NNO算法的有效性,将上述6种算法对复杂数据集进行聚类,复杂数据集包括以条状、弧形类簇为主的流形数据集,类簇样本间存在交叠情况和类簇数目超过10的数据集等. 表4列出了所选取复杂数据集的基本信息. 表5为6种聚类算法对复杂数据集聚类结果的评价指标值. 如表5所示: DPC-NNO、 IDPC-FA和FKNN-DPC算法对上述数据集的聚类效果较好; DPC-NNO算法在处理Sticks、Db、Circle3、R15、Elliptical、Forty数据集时,评价指标值均为最大值; IDPC-FA算法在Aggregation、Sticks、Forty数据集的评价指标值均为1; FKNN-DPC

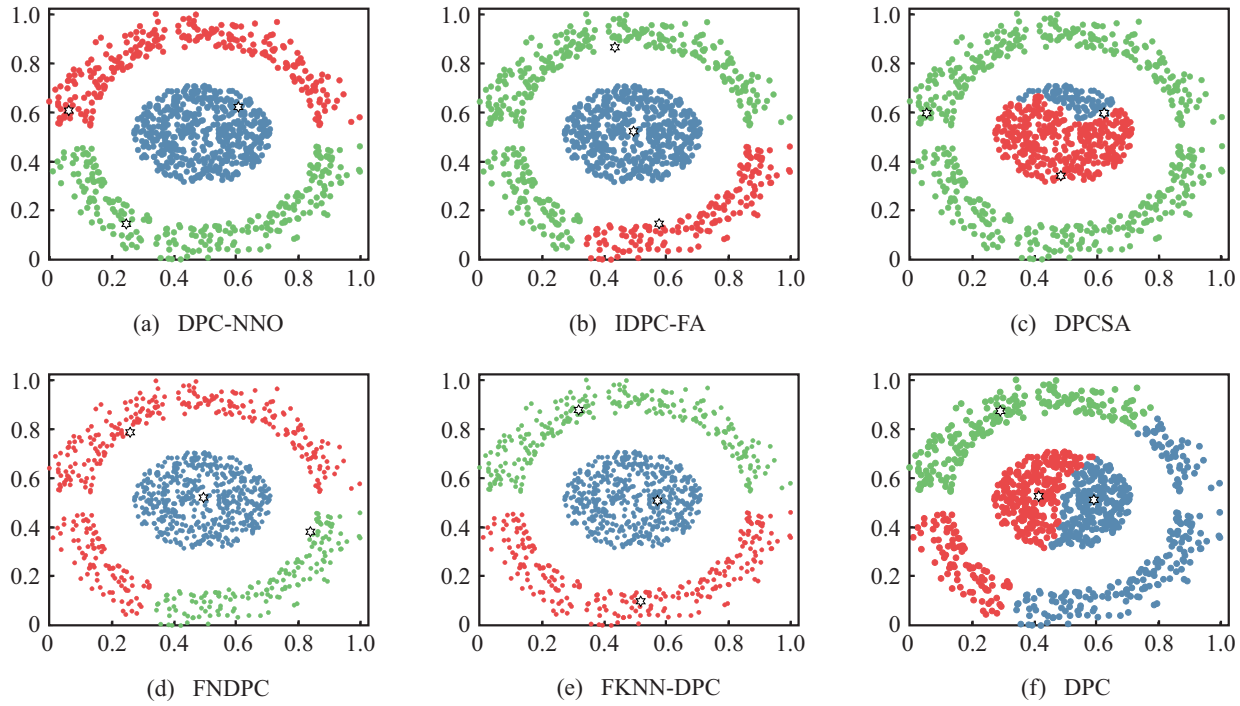


图 3　6种算法在Cmc数据集上的聚类结果

算法在Sticks、R15、D31、2d-20c、Forty、A3数据集的聚类效果好.

表 4　复杂数据集

数据集	样本规模	数据维数	类簇个数
Aggregation	788	2	7
Sticks	512	2	4
Db	630	2	4
Circle3	1 897	2	3
R15	600	2	15
D31	3 100	2	31
Elliptical	500	2	10
2d-20c	1 517	2	20
Forty	1 000	2	40
A3	7 500	2	50

3.4　UCI数据集的实验结果分析

UCI数据集包含世界上真实存在的数据,对其进行聚类可以进一步验证DPC-NNO算法的有效性. 本文将6种算法分别对8个UCI数据集进行聚类,选取的UCI数据集的基本信息如表6所示. 表7为6种算法在UCI数据集聚类结果的评价指标.

如表7所示, 对于Iris数据集,DPC-NNO、DPCSA、FKNN-DPC、FNDPC和DPC算法的聚类效果相同,IDPC-FA算法的效果稍逊一筹. 在处理Ecoli数据集时,DPC-NNO算法的聚类效果略低于IDPC-FA算法,但均优于其他算法. 对于Wine、Ionosphere、Dermatology、 Wdbc数据集,DPC-NNO算法均取得了最佳聚类效果. 总之,对于处理UCI数据集,DPC-NNO算法的聚类效果优于对比算法.

4　结　论

DPC算法在处理密度分布不均的数据时,忽略了样本的分布特性,不能正确找到稀疏类簇的类簇中心;在分配剩余样本时,易将稀疏类簇的样本分配给密集类簇. 针对上述问题,本文提出了一种面向密度分布不均数据的近邻优化密度峰值聚类算法. DPC-NNO算法在寻找类簇中心时利用逆近邻的特点,综合考虑样本k近邻和非k近邻对样本密度的影响,从而能更容易找到稀疏类簇的类簇中心;在分配剩余样本时利用共享近邻的相似度,计算样本间的相似性并构造相似度矩阵,让同一类簇的样本相似性更高,从而对稀疏类簇和密集类簇的样本进行正确分配. 实验表明,DPC-NNO算法能够很好地处理密度分布不均的数据集,并且对复杂数据集和UCI数据集的处理效果也优于对比算法. 本文对密度峰值聚类算法进行了深入研究. 未来的工作将考虑以下问题: 1) 自适应选择算法参数,本文的参数近邻数k需人为设定,下一步将研究如何让算法根据不同数据集自适应选择k值; 2) 大规模数据聚类问题,DPC-NNO算法处理人工和真实数据集时,虽有较好的聚类效果,但对大规模数据的聚类时间较长,未来也将研究如何提高算法的运行效率.

表5 6种算法在10个复杂数据集上的聚类结果评价指标值

聚类算法	Aggregation				Sticks			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NNO	0.986 6	0.992 0	0.998 3	18	**1**	**1**	**1**	2
IDPC-FA	**1**	**1**	**1**	–	**1**	**1**	**1**	–
DPCSA	0.953 7	0.958 1	0.967 3	–	0.763 4	0.636 0	0.744 3	–
FNDPC	0.986 4	0.991 3	0.993 2	0.02	**1**	**1**	**1**	0.22
FKNN-DPC	0.990 5	0.994 9	0.996 0	20	**1**	**1**	**1**	7
DPC	0.992 2	0.995 6	0.996 6	4.1	**1**	**1**	**1**	1.7

聚类算法	Db				Circle3			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NNO	**1**	**1**	**1**	9	**1**	**1**	**1**	19
IDPC-FA	0.652 6	0.503 3	0.699 9	–	0.462 9	0.438 5	0.765 2	–
DPCSA	0.413 6	0.109 6	0.468 9	–	0.295 0	0.083 3	0.524 2	–
FNDPC	0.643 1	0.441 2	0.670 0	0.74	0.423 6	0.273 2	0.586 3	0.29
FKNN-DPC	0.510 7	0.271 8	0.579 3	19	0.706 3	0.613 9	0.779 0	32
DPC	0.518 5	0.279 4	0.585 3	4	0.359 6	0.301 5	0.604 8	0.3

聚类算法	R15				D31			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NNO	**0.993 8**	**0.992 8**	**0.993 3**	17	0.963 7	0.947 9	0.951 3	43
IDPC-FA	**0.993 8**	**0.992 8**	**0.993 3**	–	0.957 5	0.940 2	0.942 1	–
DPCSA	0.988 5	0.985 7	0.986 6	–	0.955 2	0.935 3	0.937 4	–
FNDPC	**0.993 8**	**0.992 8**	**0.993 3**	0.03	0.955 5	0.936 4	0.938 5	0.04
FKNN-DPC	**0.993 8**	**0.992 8**	**0.993 3**	27	**0.965 4**	**0.952 3**	**0.953 8**	28
DPC	**0.993 8**	0.992 8	**0.993 3**	0.6	0.957 2	0.938 4	0.940 4	1.0

聚类算法	Elliptical				2d-20c			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NNO	**0.995 6**	**0.996 0**	**0.998 0**	7	0.991 4	0.991 7	0.992 2	14
IDPC-FA	0.966 8	0.970 1	0.984	–	0.991 6	0.992 1	0.995 4	–
DPCSA	0.919 7	0.800 2	0.824 8	–	0.994 3	0.994 1	0.994 4	–
FNDPC	0.979 0	0.966 8	0.970 1	0.04	0.991 3	0.991 6	0.992 1	0.06
FKNN-DPC	0.908 1	0.860 7	0.879 5	14	**0.965 9**	**0.935 3**	**0.940 4**	7
DPC	0.979 0	0.966 8	0.970 1	0.8	0.991 3	0.991 6	0.992 1	2.7

聚类算法	Forty				A3			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NNO	**1**	**1**	**1**	5	0.989 8	0.984 1	0.984 5	41
IDPC-FA	**1**	**1**	**1**	–	0.990 7	0.987 2	0.987 5	–
DPCSA	**1**	**1**	**1**	–	0.983 9	0.972 2	0.972 8	–
FNDPC	**1**	**1**	**1**	0.02	0.987 8	0.988 0	0.988 2	0.03
FKNN-DPC	0.988 0	0.974 0	0.974 9	3	**0.992 6**	**0.988 0**	**0.988 2**	22
DPC	**1**	**1**	**1**	0.2	0.987 6	0.981 4	0.981 8	0.3

表6 UCI数据集

数据集	样本规模	数据维数	类簇个数	数据集	样本规模	数据维数	类簇个数
Iris	150	4	3	Inonsphere	351	34	2
Wine	178	13	3	Libras	360	90	15
Seeds	210	7	3	Dermatology	366	33	6
Ecoli	336	8	8	Wdbc	569	30	2

表 7　6种算法在8个UCI数据集上的聚类结果评价指标值

聚类算法	Iris				Wine			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NNO	**0.883 1**	**0.903 8**	**0.935 5**	7	**0.888 5**	**0.913 4**	**0.943 5**	30
IDPC-FA	0.862 3	0.885 7	0.923 3	–	0.767 5	0.771 3	0.847 8	–
DPCSA	**0.883 1**	**0.903 8**	**0.935 5**	–	0.748 0	0.741 4	0.828 3	–
FNDPC	**0.883 1**	**0.903 8**	**0.935 5**	0.11	0.789 8	0.802 5	0.868 6	0.26
FKNN-DPC	**0.883 1**	**0.903 8**	**0.935 5**	22	0.848 1	0.883 9	0.922 9	8
DPC	**0.883 1**	**0.903 8**	**0.935 5**	3.2	0.769 5	0.770 3	0.847 4	2.4

聚类算法	Seeds				Ecoli			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NNO	0.760 8	**0.812 1**	**0.874 2**	7	0.642 5	0.691 1	0.772 5	13
IDPC-FA	0.729 9	0.767 0	0.844 4	–	**0.663 8**	**0.756 1**	**0.828 4**	–
DPCSA	0.660 9	0.687 3	0.791 8	–	0.440 6	0.459 3	0.646 7	–
FNDPC	0.713 6	0.754 5	0.836 1	0.07	0.483 3	0.561 8	0.717 8	0.35
FKNN-DPC	**0.775 7**	0.802 4	0.868 2	9	0.587 8	0.589 4	0.702 7	2
DPC	0.729 8	0.767 0	0.844 4	0.7	0.522 3	0.410 9	0.554 5	2.1

聚类算法	Inonsphere				Libras			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NN	**0.402 1**	**0.527 2**	**0.797 7**	41	0.554 6	0.340 5	0.392 0	10
IDPC-FA	0.135 5	0.218 3	0.643 2	–	0.573 3	**0.381 6**	**0.424 7**	–
DPCSA	0.133 5	0.213 5	0.639 0	–	0.538 8	0.309 5	0.379 1	–
FNDPC	0.163 0	0.248 3	0.653 1	0.06	0.549 4	0.329 0	0.386 9	0.17
FKNN-DPC	0.348 5	0.479 0	0.771 6	8	0.555 4	0.345 9	0.404 4	10
DPC	0.148 4	0.230 6	0.644 9	1.8	**0.583 2**	0.362 6	0.419 0	0.5

聚类算法	Dermatology				Wdbc			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
DPC-NNO	**0.875 9**	**0.885 3**	**0.903 8**	4	**0.724 8**	**0.817 6**	**0.916 9**	28
IDPC-FA	0.863 8	0.877 2	0.901 8	–	0.623 7	0.742 3	0.882 9	–
DPCSA	0.747 0	0.609 9	0.692 2	–	0.336 1	0.377 1	0.759 5	–
FNDPC	0.793 3	0.802 9	0.844 1	0.17	0.607 6	0.730 5	0.875 8	0.05
FKNN-DPC	0.806 6	0.836 1	0.870 9	35	0.642 3	0.761 3	0.889 4	2
DPC	0.835 4	0.838 9	0.871 5	1.5	0.637 5	0.754 8	0.887 6	0.7

参考文献(References)

[1] Jia R, Khadka A, Kim I. Traffic crash analysis with point-of-interest spatial clustering[J]. Accident Analysis & Prevention, 2018, 121: 223-230.

[2] Liao H J, Lin C H R, Lin Y C, et al. Intrusion detection system: A comprehensive review[J]. Journal of Network and Computer Applications, 2013, 36(1): 16-24.

[3] Pei H X, Zheng Z R, Wang C, et al. D-FCM: Density based fuzzy *c*-means clustering algorithm with application in medical image segmentation[J]. Procedia Computer Science, 2017, 122: 407-414.

[4] Lei T, Liu P, Jia X H, et al. Automatic fuzzy clustering framework for image segmentation[J]. IEEE Transactions on Fuzzy Systems, 2019, 28(9): 2078-2092.

[5] Nizar A H, Dong Z Y, Wang Y. Power utility nontechnical loss analysis with extreme learning machine method[J]. IEEE Transactions on Power Systems, 2008, 23(3): 946-955.

[6] 张曦, 李璠, 付雪峰, 等. 随机学习萤火虫算法优化的模糊软子空间聚类算法[J]. 江西师范大学学报: 自然科学版, 2021, 45(2): 137-144.
(Zhang X, Li F, Fu X F, et al. The fuzzy soft subspace clustering algorithm optimized by random learning firefly algorithm[J]. Journal of Jiangxi Normal University: Natural Science Edition, 2021, 45(2): 137-144.)

[7] Likas A, Vlassis N, Verbeek J J. The global k-means clustering algorithm[J]. Pattern Recognition, 2003, 36(2): 451-461.

[8] Gurrutxaga I, Albisua I, Arbelaitz O, et al. SEP/COP: An efficient method to find the best partition in hierarchical clustering based on a new cluster validity index[J]. Pattern Recognition, 2010, 43(10): 3364-3373.

[9] Ester M, Kriegel H P, Sander J, et al. A density-based algorithm for discovering clusters in large spatial databases with noise[C]. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining. New York: ACM, 1996: 226-231.

[10] Wang W, Yang J, Muntz R R. STING: A statistical information grid approach to spatial data mining[C]. Proceedings of the 23rd International Conference on Very Large Data Bases. New York: ACM, 1997: 186-195.

[11] Hartigan J A, Wong M A. A K-means clustering algorithm[J]. Journal of the Royal Statistical Society Series C: Applied Statistics, 1979, 28(1): 100-108.

[12] Park H S, Jun C H. A simple and fast algorithm for K-medoids clustering[J]. Expert Systems with Applications, 2009, 36(2): 3336-3341.

[13] Ester M, Kriegel H P, Sander J, et al. A density-based algorithm for discovering clusters in large spatial databases with noise[C]. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining. New York: ACM, 1996: 226-231.

[14] Rodriguez A, Laio A. Clustering by fast search and find of density peaks[J]. Science, 2014, 344(6191): 1492-1496.

[15] 吴润秀, 尹士豪, 赵嘉, 等. 基于相对密度估计和多簇合并的密度峰值聚类算法[J]. 控制与决策, 2023, 38(4): 1047-1055.
(Wu R X, Yin S H, Zhao J, et al. Density peaks clustering based on relative density estimating and multi cluster merging[J]. Control and Decision, 2023, 38(4): 1047-1055.)

[16] Yuan X N, Yu H, Liang J, et al. A novel density peaks clustering algorithm based on K nearest neighbors with adaptive merging strategy[J]. International Journal of Machine Learning and Cybernetics, 2021, 12(10): 2825-2841.

[17] Cheng D D, Zhu Q S, Huang J L, et al. Clustering with local density peaks-based minimum spanning tree[J]. IEEE Transactions on Knowledge and Data Engineering, 2021, 33(2): 374-387.

[18] Tao X M, Guo W J, Ren C, et al. Density peak clustering using global and local consistency adjustable manifold distance[J]. Information Sciences, 2021, 577: 769-804.

[19] Cheng D D, Huang J L, Zhang S L, et al. Improved density peaks clustering based on shared-neighbors of local cores for manifold data sets[J]. IEEE Access, 2019, 7: 151339-151349.

[20] 赵嘉, 姚占峰, 吕莉, 等. 基于相互邻近度的密度峰值聚类算法[J]. 控制与决策, 2021, 36(3): 543-552.
(Zhao J, Yao Z F, Lv L, et al. Density peaks clustering based on mutual neighbor degree[J]. Control and Decision, 2021, 36(3): 543-552.)

[21] Yu D H, Liu G J, Guo M Z, et al. Density peaks clustering based on weighted local density sequence and nearest neighbor assignment[J]. IEEE Access, 2019, 7: 34301-34317.

[22] 孙林, 秦小营, 徐久成, 等. 基于K近邻和优化分配策略的密度峰值聚类算法[J]. 软件学报, 2022, 33(4): 1390-1411.
(Sun L, Qin X Y, Xu J C, et al. Density peak clustering algorithm based on K-nearest neighbors and optimized allocation strategy[J]. Journal of Software, 2022, 33(4): 1390-1411.)

[23] Yu H, Chen L Y, Yao J T. A three-way density peak clustering method based on evidence theory[J]. Knowledge-Based Systems, 2021, 211: 106532.

[24] Zhao J, Tang J J, Shi A Y, et al. Improved density peaks clustering based on firefly algorithm[J]. International Journal of Bio-Inspired Computation, 2020, 15(1): 24-42.

[25] Du M J, Ding S F, Xue Y. A robust density peaks clustering algorithm using fuzzy neighborhood[J]. International Journal of Machine Learning and Cybernetics, 2018, 9(7): 1131-1140.

[26] Xie J Y, Gao H C, Xie W X, et al. Robust clustering by detecting density peaks and assigning points based on fuzzy weighted K-nearest neighbors[J]. Information Sciences, 2016, 354: 19-40.

[27] Vinh N, Epps J, Bailey J. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance[J]. Journal of Machine Learning Research, 2010, 11(1): 2837-2854.

[28] Fowlkes E B, Mallows C L. A method for comparing two hierarchical clusterings[J]. Journal of the American Statistical Association, 1983, 78(383): 553-569.

作者简介

陈蔚昌(2000−), 男, 硕士生, 从事数据挖掘的研究, E-mail: chenweichang2000@163.com;

赵嘉(1981−), 男, 教授, 博士, 从事机器学习与数据挖掘、计算智能等研究, E-mail: zhaojia925@163.com;

肖人彬(1965−), 男, 教授, 博士生导师, 从事复杂系统建模与分析、群集智能等研究, E-mail: rbxiao@163.com;

王晖(1982−), 男, 教授, 博士, 从事群智能算法、进化计算等研究, E-mail: huiwang@whu.edu.cn;

崔志华(1976−), 男, 教授, 博士生导师, 从事大数据建模与优化、区块链等研究, E-mail: cuizhihua@tyust.edu.cn.