文章编号:1672-3961(2025)02-0058-13 DOI:10.6040/j.issn.1672-3961.0.2024.045

融合局部截断距离及小簇合并的密度峰值聚类

陈素根[1,2],赵志忠[1*]
(1.安庆师范大学数理学院, 安徽 安庆 246133; 2.安徽省大别山区域复杂生态系统建模、仿真与控制重点实验室, 安徽 安庆 246133)

摘要:针对密度峰值聚类算法定义的截断距离仅考虑样本全局分布,在样本分配时容易产生“多米诺骨牌”现象等问题,提出一种融合局部截断距离及小簇合并的密度峰值聚类算法。基于样本局部分布信息计算每个样本截断距离和局部密度,有利于准确获得复杂结构数据集上密度峰;根据样本决策值之间差值关系选择潜在密度峰并形成多个小簇;定义一种新的小簇间相似度,根据此相似度将小簇合并获得聚类结果,有效避免了“多米诺骨牌”现象。采用 6 个人工数据集和 8 个 UCI 数据集进行验证,所提算法在上述 14 个数据集上的标准化互信息、调整兰德系数和调整互信息平均值比 5 个对比算法平均提高 18.15%、28.99%和 20.22%,比原始密度峰值聚类算法提高 30.06%,47.15%和 31.90%,具有较好的聚类效果。

关键词:聚类;密度峰值聚类;截断距离;局部密度;潜在密度峰
中图分类号:TP391 **文献标志码**:A
引用格式:陈素根,赵志忠. 融合局部截断距离及小簇合并的密度峰值聚类[J]. 山东大学学报(工学版),2025,55(2):58-70.
CHEN Sugen, ZHAO Zhizhong. Density peak clustering combining local truncation distance and small clusters merging[J]. Journal of Shandong University (Engineering Science), 2025, 55(2):58-70.

Density peak clustering combining local truncation distance and small clusters merging

CHEN Sugen[1,2], ZHAO Zhizhong[1*]
(1. School of Mathematics and Physics, Anqing Normal University, Anqing 246133, Anhui, China; 2. Key Laboratory of Modeling, Simulation and Control of Complex Ecosystem in Dabie Mountains of Anhui Higher Education Institutes, Anqing 246133, Anhui, China)

Abstract: Aiming at the problems that the truncation distance defined by the density peak clustering algorithm only considered the global distribution of samples and the "domino" phenomenon was easy to occur when assigning samples, a novel density peak clustering algorithm combining local truncation distance and small clusters merging was proposed. The truncation distance and local density of each sample were calculated based on the local distribution information of samples, which were conducive to accurately obtaining the density peaks on complex structure datasets. Potential density peaks were selected based on the difference between samples decision values and multiple small clusters were formed. A new kind of similarity between clusters was defined, and clusters were merged to obtain clustering results according to this similarity, which effectively avoided the "domino" phenomenon. Compared with several clustering algorithms on six synthetic datasets and eight UCI datasets, the standardized mutual information, adjusted rand index and adjusted mutual information average values of the proposed algorithm on 14 datasets were 18.15%, 28.99% and 20.22% higher than the five comparison algorithms on average, especially 30.06%, 47.15% and 31.90% higher than original density peak clustering algorithm. Experimental results showed the proposed algorithm had a good clustering effect.
Keywords: clustering; density peak clustering; truncation distance; local density; potential density peaks

收稿日期:2024-02-29
基金项目:国家自然科学基金青年基金资助项目(61702012);安徽省自然科学基金面上资助项目(2008085MF193);安徽省高等学校科学研究重点资助项目(2024AH051095)
第一作者简介:陈素根(1982—),男,安徽当涂人,教授,硕士生导师,博士,主要研究方向为模式识别、机器学习等。E-mail: chensugen@126.com
*通信作者简介:赵志忠(1999—),男,安徽淮南人,硕士研究生,主要研究方向为模式识别、机器学习。E-mail: zhaozz1999@126.com

0　引言

随着大数据时代到来,聚类作为一种能够发掘数据内部潜在信息的数据挖掘方法,已经广泛应用于图像处理[1]、经济分析[2]、生物医学[3]、模式识别[4]和社区检测[5]等领域。根据对数据处理方式的不同,聚类算法包括多种类型,如:基于划分的 *k*-means 算法[6]、基于网格的 STING(statistical information grid)算法[7]、基于层次的 BRICH(balanced iterative reducing and clustering using hierarchies)算法[8]和基于密度的 DBSCAN 算法[9]等,更多类型可见文献[10-12]。*k*-means 算法通过优化目标函数使样本到聚类中心距离平方和最小,该算法易于实现且速度较快,在球形类簇上效果较好,在非球形类簇上效果较差;STING 算法将样本划分到不同网格中,利用网格关系进行聚类,有效降低了时间复杂度,该算法受底层网格影响较大,对于变密度数据集聚类准确率较低;BRICH 算法基于聚类特征表示各个聚类层次,采用自下而上策略合并样本完成聚类,该算法聚类速度较快,对参数敏感且在非凸数据集上聚类效果较差。上述类型聚类算法在面对任意形状和变密度数据聚类问题时,聚类结果往往不尽人意。DBSCAN 作为一种经典密度聚类算法,可以通过样本分布紧密性识别任意形状类簇,该算法需要输入较多参数且对参数敏感。文献[13]提出了一种新的密度聚类算法:密度峰值聚类(density peak clustering, DPC)。DPC 算法基于两点假设:第一是密度峰的局部密度要明显大于周围样本的局部密度,第二是任意两个密度峰之间相距较远。该算法结合样本密度和距离关系进行聚类,其原理简单、聚类过程无需迭代、输入参数少且可识别任意形状类簇,受到广泛关注和研究[14-16]。DPC 算法也存在着一些缺点,如:采用全局截断距离定义密度,这种方式只考虑了样本全局分布,忽略了局部分布,不能准确刻画变密度数据集低密度簇中分布较集中样本的密度,容易在高密度簇中产生多个密度峰,导致算法对变密度数据集聚类效果不佳;分配策略容易产生“多米诺骨牌”现象,即高密度样本错误分配会导致后续低密度样本都分配错误,引发分配错误扩大化。

近年来,众多学者就 DPC 算法缺点提出了不同改进策略。在局部密度改进方面,文献[17]提出了基于改进相似度和分配策略的 DPC 算法(density peaks clustering algorithm based on improved similarity and allocation strategy, DPCV),该算法将样本方差引入密度定义中,减少变密度数据集类簇间的密度差。文献[18]提出了基于核心点识别和 *k* 近邻核密度估计的鲁棒聚类算法,该算法利用 *k* 近邻计算样本密度,形成代表团进行聚类。文献[19]提出了自适应最近邻 DPC 算法,该算法引入样本自适应邻居来精确定义样本密度,从而获得低密度簇的密度峰。文献[20]提出基于加权局部密度序列和最近邻分配的 DPC 算法(density peaks clustering based on weighted local density sequence and nearest neighbor assignment, DPCSA),该算法考虑 *k* 近邻内外样本对密度的贡献,定义新的局部密度。文献[21-22]将模糊领域和样本与其近邻间相对关系引入密度计算中,这些算法都在一定程度上改进了密度计算方式。在样本分配策略改进方面,文献[23]提出了基于关联度转移方法的局部密度峰快速分层聚类算法(fast hierarchical clustering of local density peaks via an association degree transfer method, FHC-LDP),该算法根据密度峰将数据集划分成不同子簇,利用层次聚类合并子簇。文献[24]提出了基于连通性估计的 DPC 算法(density peak clustering with connectivity estimation, DPC-CE),该算法利用图的连通性改进了样本分配中相对距离计算策略。文献[25]提出了具有模糊连通性自适应两阶段密度聚类算法,该算法结合 DPC 和 DBSCAN 算法优势,在确定密度峰和样本分配隶属度时同时考虑距离和样本间模糊连通性,提高了聚类效果。文献[26]提出了面向流形数据共享近邻 DPC 算法,该算法由样本间近邻关系定义样本间相似度,基于此相似度分配样本。文献[27-28]通过引入最小生成树和最小生成森林策略提高 DPC 算法分配准确性。

综上所述,虽然改进的 DPC 算法在一定程度上提高了原始 DPC 算法的聚类性能,部分算法仍是从全局分布角度改进样本密度的计算方式和分配策略,所定义的相似度导致这些算法在处理类簇间存在交叉样本复杂结构数据集时聚类效果不佳。为解决上述问题,本研究提出一种融合局部截断距离及小簇合并的密度峰值聚类算法(density peak clustering combining local truncation distance and small clusters merging, LSDPC),该算法基于局部截断距离定义了新的样本密度,利用样本决策值间差值关系获得潜在密度峰并生成多个小簇,定义一种新的簇间相似度,按照该相似度将小簇合并获得聚类结果。

1 DPC 算法及缺点分析

本章将简要介绍 DPC 算法相关概念、聚类流程以及举例分析该算法缺点。

1.1 DPC 算法

对于数据集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_n\}$ 中任意样本 $\boldsymbol{x}_i$，其局部密度 ρ_i 为

$$\rho_i=\sum_{j\neq i}\chi(d_{ij}-d_c),\ \chi(u)=\begin{cases}1, & u<0\\ 0, & \text{otherwise}\end{cases}, \tag{1}$$

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

式中：d_{ij} 为样本间欧氏距离；d_c 为截断距离，这是 DPC 算法需要给定的参数，按照 DPC 算法要求，d_c 选取要使每个样本周围样本平均数占总样本数的 1%~2%。DPC 算法有两种密度计算方式，当聚类大规模数据集时，使用式(1)截断核获得样本密度；否则，使用式(2)高斯核获得样本密度。

DPC 算法由样本间密度关系获得相对距离：

$$\delta_i=\begin{cases}\min\limits_{j:\rho_j>\rho_i} d_{ij}, & \text{if } \exists j \text{ s.t. } \rho_j>\rho_i\\ \max\limits_{j} d_{ij}, & \text{otherwise}\end{cases}。 \tag{3}$$

当求得所有样本的密度和相对距离后，以密度为 x 轴，相对距离为 y 轴建立决策图，选择位于图中右上角点作为密度峰。DPC 算法还可以利用样本决策值 γ_i 选择密度峰：

$$\gamma_i=\rho_i\times\delta_i。 \tag{4}$$

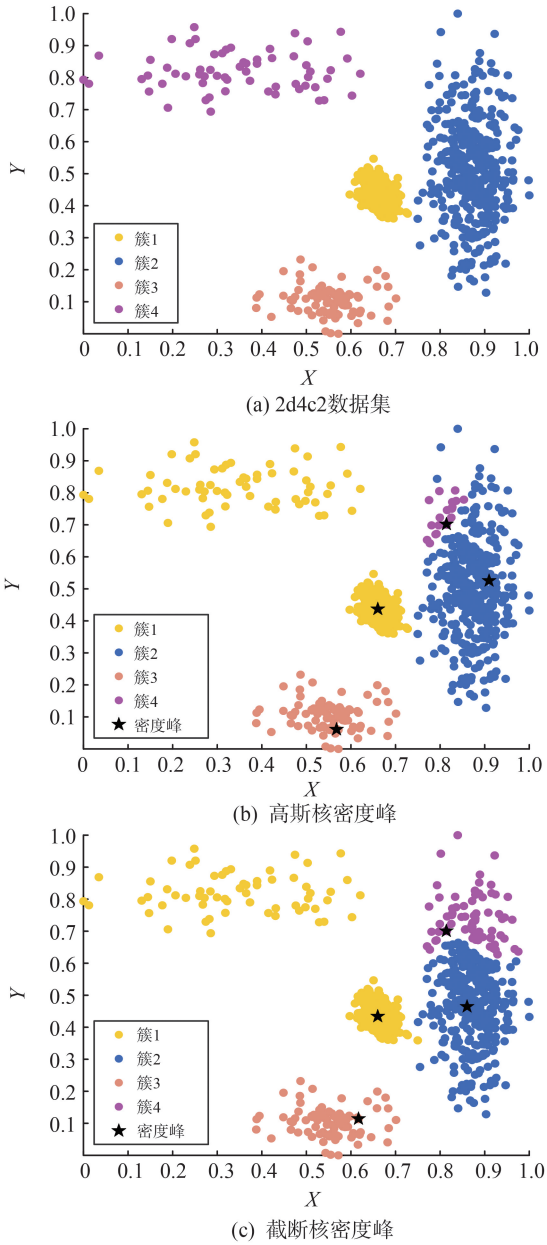
利用决策值选择密度峰的具体做法为，将所有样本决策值降序排列，选择决策值较大样本为密度峰。分配阶段是给密度峰赋予不同类标签，将剩余样本标签标记为比其密度大且离其最近样本所属标签，完成聚类。

1.2 DPC 算法缺点分析

DPC 算法缺点主要在以下两个方面：

(1) DPC 算法利用样本全局分布信息计算截断距离，由此获得样本密度，该密度无法有效处理变密度数据集，使得 DPC 算法往往会在变密度数据集某个高密度类簇中选择多个样本作为密度峰，导致后续样本分配过程出错。

以 2d4c2 数据集[14]为例，叙述 DPC 算法上述缺点。分别绘制 2d4c2 数据集的原始分布图、DPC 算法采用高斯核密度和截断核密度在 2d4c2 数据集上找到的密度峰，如图 1 所示。



(a) 2d4c2数据集

(b) 高斯核密度峰

(c) 截断核密度峰

图 1 DPC 算法在 2d4c2 数据集上密度峰

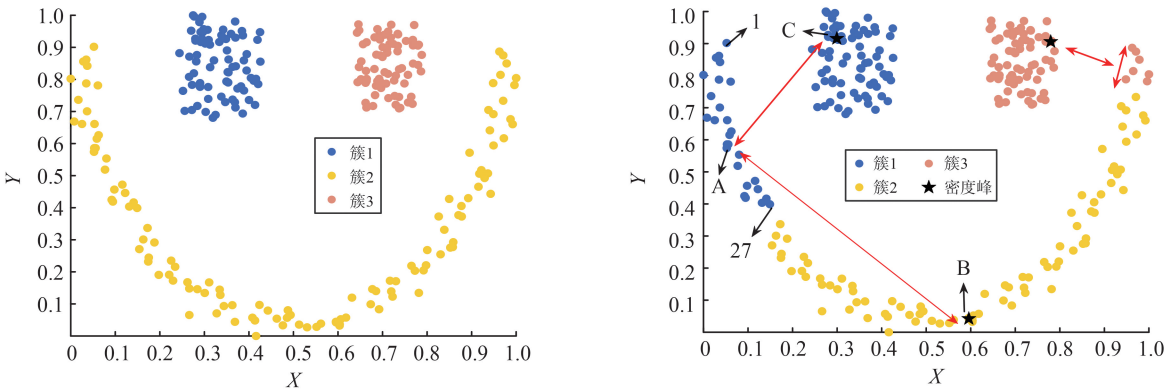
Fig.1 Density peaks obtained on 2d4c2 by DPC

从图 1(a)可以发现，2d4c2 数据集所包含 4 个类簇密度不统一，位于左上方簇密度要明显低于剩下几个簇密度，是一个典型的变密度数据集。DPC 算法两种密度定义方式均依赖于截断距离，该截断距离只考虑样本全局分布，在变密度数据集中按照 DPC 算法截断距离计算方式获得截断距离值，由此计算样本密度会导致高密度簇中样本密度总体上远大于低密度簇中样本密度，不能准确刻画低密度簇中分布较集中样本密度，引发密度峰错误选择。如图 1(b)和图 1(c)所示，无论是使用高斯核密度还是截断核密度都在右方高密度簇中选取了两个密度峰，无法在左上角低密度簇中选取到密度峰，使得 DPC 算法在 2d4c2 数据集上获得了较差的聚

类结果。

(2) DPC 算法的样本分配策略容易引发“多米诺骨牌”现象,密度较高样本错误分配会引起后续密度较低样本均分配错误。

以 Lineblobs 数据集[14] 为例,叙述“多米诺现象”发生机制。绘制 Lineblobs 原始分布图和 DPC 算法采用高斯核密度在 Lineblobs 数据集上聚类结果如图 2 所示。从图 2(a)可以发现,Lineblobs 数据集上方两簇密度要明显高于下方类簇密度,同时下方簇呈流形分布,该数据集既是变密度数据集又是流形数据集。图 2(b)为 DPC 算法采用高斯核密度所得聚类结果,黑色五角星代表密度峰。由图 2(b)可知,样本 A 是样本 1 到样本 27 中密度最大值样本,样本 B 与样本 A 属于相同类簇,样本 C 与样本 A 属于不同类簇,样本 B 和 C 密度均大于样本 A 的密度。样本 A 与样本 C 之间距离小于样本 A 与样本 B 之间距离,按照 DPC 算法分配策略导致样本 A 被错误分配到样本 C 所在类簇,引起样本 1 到样本 27 全部被分配错误,这就是 DPC 算法的“多米诺骨牌”现象。位于流形类簇右上角的红色样本也是由于上述原因而被错误分配。若采取截断核计算样本密度,也会出现类似现象,这里就不再赘述了。



(a) Lineblobs数据集　　(b) 高斯核聚类结果

图 2　DPC 算法在 Lineblobs 数据集上聚类结果

Fig.2　Clustering result of DPC on Lineblobs

2　LSDPC 算法

为了解决 DPC 算法无法有效识别变密度数据集,在分配过程容易发生“多米诺骨牌”现象的问题,本研究将样本局部分布信息添加到密度计算中,由决策值间差值关系产生多个潜在密度峰,形成多个小簇,根据小簇间局部关系定义簇间相似度,将小簇合并完成聚类。图 3 给出了所提 LSDPC 算法框架图。

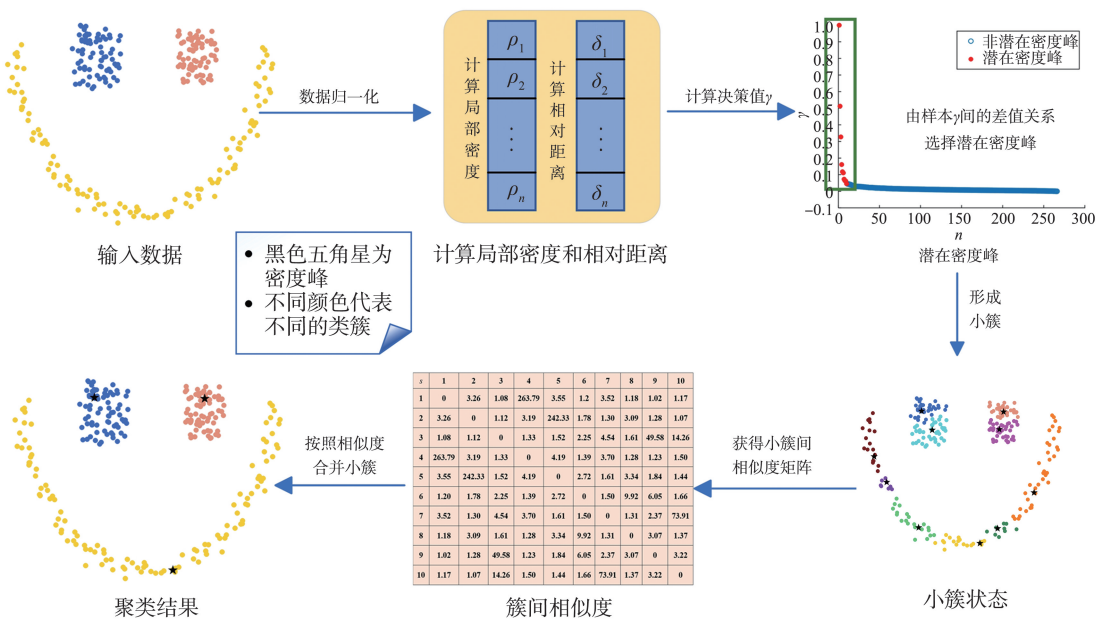


图 3　LSDPC 算法框架

Fig.3　Framework of LSDPC algorithm

2.1 基于局部截断距离的密度计算

定义 1 局部截断距离。对于数据集 D 中任意样本 $\boldsymbol{x}_i$,其截断距离 $d_c(\boldsymbol{x}_i)$ 为

$$d_c(\boldsymbol{x}_i)=\frac{\alpha\cdot(d_{\max}-d_{i1})}{\sqrt{n-2}},\tag{5}$$

式中:$d_{\max}$ 为样本间距离最大值;d_{i1} 为样本 $\boldsymbol{x}_i$ 与其最近邻样本之间的距离;n 为样本总数;α 为调控因子,用来调节截断距离大小。

式(5)从样本局部分布角度重新定义了截断距离,分子为距离差值,这在一定程度上反映了样本所处区域局部分布情况,α 和分母用来控制截断距离大小。若截断距离过大可能会将其余类样本纳入到某类样本密度计算中,导致算法无法正确处理一些结构复杂数据集;若截断距离过小可能会导致算法无法正确反映数据内部分布情况。通过试验分析知,当 α 为 1~6 时,可获得较好聚类结果。由式(5)可得,倘若样本 $\boldsymbol{x}_i$ 位于稀疏区域,则该样本与其最近邻样本之间距离较大,利用式(5)就可得到较小的 d_c;反之,位于密集区域样本会有相对较大的 d_c。式(5)根据样本所属区域局部分布情况,为每个样本赋予不同截断距离,这使得在低密度类簇内分布较集中样本也可以由其局部性质获得较大截断距离,以便在后续密度计算中突出这些样本的局部密度值。

定义 2 样本的截断近邻。对于数据集 D 中任意样本 $\boldsymbol{x}_i$,根据 $\boldsymbol{x}_i$ 截断距离 $d_c(\boldsymbol{x}_i)$,定义 $\boldsymbol{x}_i$ 截断近邻

$$T_{\mathrm{NN}}(\boldsymbol{x}_i)=\{\boldsymbol{x}_j\mid d_{ij}\leqslant d_c(\boldsymbol{x}_i)\}。\tag{6}$$

若样本 $\boldsymbol{x}_i$ 位于密集区域,其截断距离较大,就会有较多截断近邻样本;反之,位于稀疏区域的样本截断距离较小,其截断近邻样本数目也较少。若样本截断近邻样本数为 0,通常该样本为边界样本或离群样本,在后续步骤中将被排除在核心样本集之外。

定义 3 样本的局部密度。对于数据集 D 中任意样本 $\boldsymbol{x}_i$,根据 $\boldsymbol{x}_i$ 截断距离 $d_c(\boldsymbol{x}_i)$ 和截断近邻 $T_{\mathrm{NN}}(\boldsymbol{x}_i)$,得到 $\boldsymbol{x}_i$ 局部密度:

$$\rho_i=\begin{cases}\displaystyle\sum_{\boldsymbol{x}_j\in T_{\mathrm{NN}}(\boldsymbol{x}_i)}\exp\left(-\frac{d_{ij}}{d_c(\boldsymbol{x}_i)}\right), & \mid T_{\mathrm{NN}}(\boldsymbol{x}_i)\mid\neq 0\\ 0, & \mid T_{\mathrm{NN}}(\boldsymbol{x}_i)\mid=0\end{cases},\tag{7}$$

式中 $\mid T_{\mathrm{NN}}(\boldsymbol{x}_i)\mid$ 为样本 $\boldsymbol{x}_i$ 的截断近邻样本数。

通过式(7)可知,样本 $\boldsymbol{x}_i$ 截断近邻样本数越多,$\boldsymbol{x}_i$ 与其截断近邻中样本越接近,样本截断距离值越大,样本局部密度也就越大。当样本截断近邻样本数为 0 时,样本局部密度也为 0。式(7)利用样本局部分布信息,使得算法在低密度簇中也能找到密度峰,可以解决原始 DPC 无法有效处理变密度数据集问题。

为了体现式(7)中的局部密度可以更好处理变密度数据集,图 4 给出了其在 2d4c2 数据集上所得密度峰。图中样本分配策略仍采用原始 DPC 算法分配策略,黑色五角星为各个类簇的密度峰。

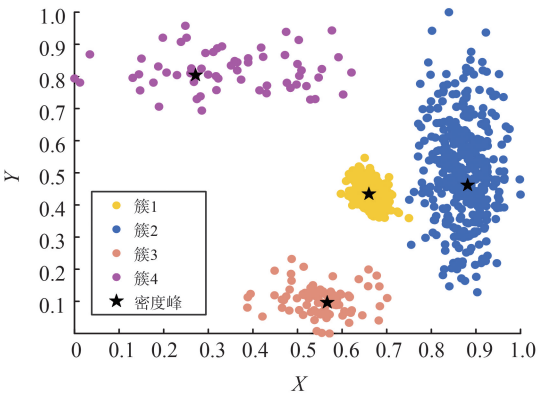


图 4 2d4c2 数据集上密度峰

Fig.4 Density peaks obtained on 2d4c2

从图 4 中可以看出,相较于图 1(b)和图 1(c)中原始 DPC 算法找到的密度峰,新定义的局部密度根据样本所属区域局部分布情况为每个样本计算不同截断距离,这可以突出低密度簇中分布较集中的样本密度,在 2d4c2 数据集的 4 个类簇上都获得了正确密度峰,取得了较好聚类结果。

为进一步展现式(7)中密度计算方式优势,图 5 给出了 LSDPC 算法和 DPC 算法在 2d4c2 数据集上所得聚类决策图。

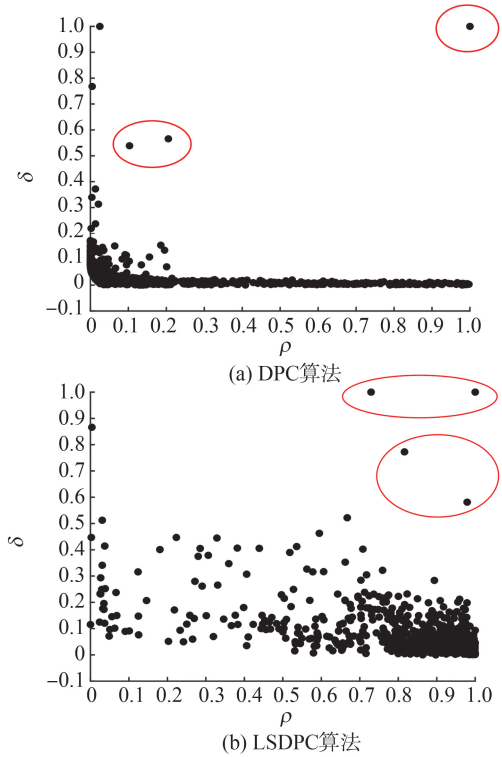


图 5 2d4c2 数据集上聚类决策图

Fig.5 Decision graph on 2d4c2

图 5(a)和图 5(b)中被圈中的点为各算法找到的密度峰。从图 5(a)可以看出,DPC 算法在 2d4c2 数据集上只能勉强找到 3 个密度峰,无法正确分辨第 4 个密度峰的位置,且各密度峰在决策图上分布比较离散;反观图 5(b),LSDPC 算法成功选取到了 4 个密度峰,这些密度峰基本位于决策图右上角,分布比较集中,易于选取。新定义的局部密度在变密度数据集低密度类簇中也可以准确找到密度峰,提升了原始 DPC 算法对复杂结构数据集处理能力。

2.2　小簇合并策略

针对 DPC 算法分配策略容易产生“多米诺骨牌”现象,提出一种小簇合并策略。根据样本决策值选择潜在密度峰。通过大量数据集试验发现,各数据集中大部分样本决策值很低,只有极少数样本决策值较高,若将样本决策值降序排列并绘制在图中,可以发现决策值较高样本零星分布于图中左上角,与后续决策值大小差异明显。基于以上分析,计算决策值差值序列,在此基础上给出差值序列平均值定义,由差值序列与平均值关系选择潜在密度峰,具体定义如下。

定义 4　决策值差值序列。对于降序排列后决策值,分别计算相邻两个决策值的差,定义决策值差值序列

$$S=\{\lambda_i \mid \lambda_i=\gamma_{\mathrm{sd}}(\boldsymbol{x}'_i)-\gamma_{\mathrm{sd}}(\boldsymbol{x}'_{i+1}),\ i=1,\cdots,n-1\},\tag{8}$$

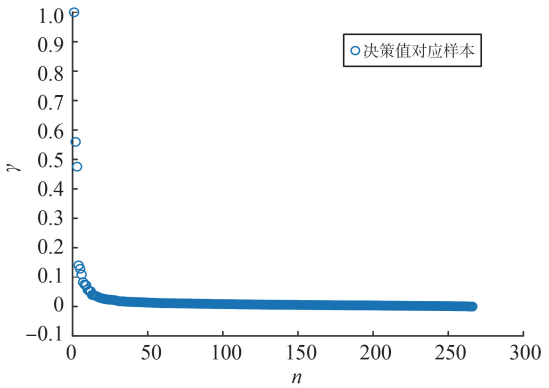
式中,$\boldsymbol{x}'_i$为决策值降序排列后处于第 i 位的决策值所对应的样本,$\gamma_{\mathrm{sd}}(\boldsymbol{x}'_i)$为样本 $\boldsymbol{x}'_i$的决策值,λ_i 是样本 $\boldsymbol{x}'_i$与 $\boldsymbol{x}'_{i+1}$的决策值之差。

定义 5　决策值差值序列平均值。对于决策值差值序列 S,定义序列平均值 λ_{aver}如下:

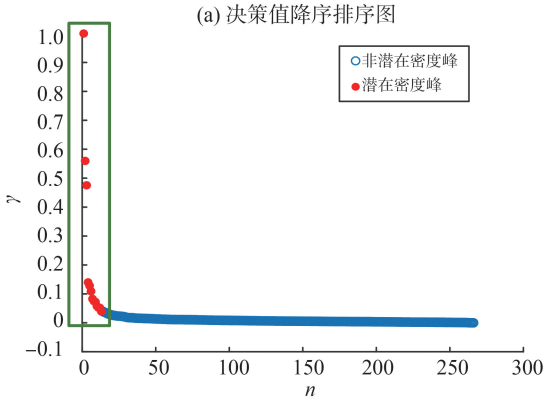
$$\lambda_{\mathrm{aver}}=\frac{1}{n-1}\sum_{i=1}^{n-1}\lambda_i。\tag{9}$$

定义 6　潜在密度峰。将式(8)中决策值差值大于式(9)所定义平均值的样本称为潜在密度峰,把这些样本构成的集合记作 P_{eaks}。

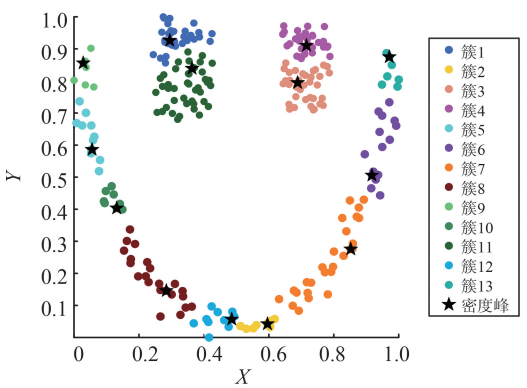
以 Lineblobs 数据集为例,绘制降序排列样本决策值,如图 6(a)所示。图中只有零散样本决策值较高,后续样本决策值都比较低且呈线性分布。将决策值降序排列,依次计算前后两个决策值之差,获得差值序列,将决策值之差大于平均差值的样本作为潜在密度峰。经试验验证,此方法所得潜在密度峰数目一般大于数据集真实类簇数。如图 6(b)所示,绿框中红色点为 LSDPC 算法在 Lineblobs 数据集上获得的 13 个潜在密度峰。当潜在密度峰集合 P_{eaks}获得后,按照原始 DPC 算法分配策略将剩余样本聚成多个小簇。如图 6(c)所示,是在 Lineblobs 数据集上所得 13 个小簇。图中不同颜色代表不同小簇,黑色五角星是各簇的密度峰。



(a) 决策值降序排序图



(b) 潜在密度峰



(c) 采用高斯核生成小簇

图 6　小簇生成示意图

Fig.6　Illustration of generating small clusters

当小簇获得以后,可根据小簇间相互关系定义簇间相似度。

定义 7　簇内平均密度。对于任意小簇 C_i,定义小簇的簇内平均密度

$$\rho_{C_i}=\frac{1}{|C_i|}\sum_{\boldsymbol{x}_m\in C_i}\rho_m,\tag{10}$$

式中$|C_i|$为小簇 C_i 中的样本数目。

定义 8　簇间距离。对于任意两个小簇 C_i 和 C_j,定义小簇间距离

$$d_{\mathrm{between}}(C_i,C_j)=\min_{\boldsymbol{x}_m\in K(C_i),\,\boldsymbol{x}_n\in K(C_j)}d_{mn},\tag{11}$$

式中 $K(C_i)=\{\boldsymbol{x}_m \mid \rho_m \geqslant 0.9\times\rho_{C_i}\}$ 为小簇 C_i 的核心样本集。

通过式(11)可以发现，小簇间距离定义为两个小簇核心样本集间最短距离，该距离越小，说明两个小簇越接近，属于同一类概率就越大。核心样本集利用密度关系排除了各个小簇中密度较小离群样本或不同类簇间交叉样本对簇间距离计算影响。

定义 9　小簇间交叉样本对。对于任意两个小簇 C_i 和 C_j，定义小簇间交叉样本对

$$P(C_i,C_j)=\{(\boldsymbol{x}_m,\boldsymbol{x}_n)\mid \boldsymbol{x}_m\in K(C_i)\cap T_{\mathrm{NN}}(\boldsymbol{x}_n),\ \boldsymbol{x}_n\in K(C_j)\cap T_{\mathrm{NN}}(\boldsymbol{x}_m)\},\tag{12}$$

式中$(\boldsymbol{x}_m,\ \boldsymbol{x}_n)$为小簇 C_i 和 C_j 间一对交叉样本。

由式(12)可以看出，小簇间交叉样本越多，两个小簇联系就越紧密，同属一类的概率也就越大。某些算法[23]将式(12)中核心样本集 $K(C_i)$ 和 $K(C_j)$ 替换成小簇 C_i 和 C_j 从而获得小簇间交叉样本，这些交叉样本在原始数据中可能属于不同的类(异类小簇交叉样本)，利用核心样本集获得小簇间交叉样本可以减轻上述异类小簇交叉样本对聚类结果的影响。反观在原始数据中本身属于同一类的小簇，考虑小簇间交叉样本(同类小簇交叉样本)，这些交叉样本通常位于该类内部区域，故其密度高于该类边界区域样本，根据核心样本集的定义可知这些样本一般存在于核心样本集中，不会像异类小簇交叉样本一样被排除在核心样本集之外。由式(12)获得的小簇间交叉样本对数目，可以体现同类小簇间紧密程度。图 7 给出了存在相互交叉样本数据集示例。

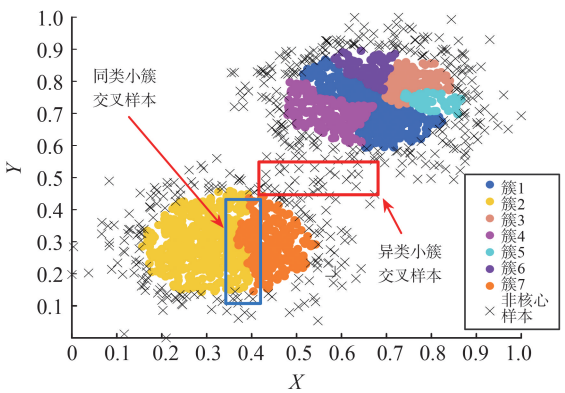


图 7　小簇间的交叉样本

Fig.7　Cross samples between small clusters

图 7 中数据集由 2 类组成，图中不同颜色代表不同小簇核心样本集，黑叉为各小簇的非核心样本。蓝筐圈中样本为同类小簇间交叉样本，即式(12)中定义的交叉样本，红框圈中样本为异类小簇间交叉样本，这些样本是将式(12)中的核心样本集 $K(C_i)$ 和 $K(C_j)$ 直接替换成小簇 C_i 和 C_j 获得的交叉样本。可以发现，式(12)利用核心样本集，保留了同类小簇之间交叉样本，排除了异类小簇间交叉样本对小簇间相互关系的影响。

定义 10　簇间密度关系值。对于任意两个小簇 C_i 和 C_j，定义簇间密度关系值

$$r_{ij}=\frac{2\sqrt{\rho_{C_i}\times\rho_{C_j}}}{\rho_{C_i}+\rho_{C_j}}。\tag{13}$$

簇内平均密度反映了该小簇整体分布情况，属于同一类小簇应具有相同的分布，簇内平均密度比较接近。由式(13)知，两个小簇平均密度越接近，密度关系值就越大，越可能属于同一类。

定义 11　小簇间相似度。对于任意两个小簇 C_i 和 C_j，定义小簇间相似度

$$s(C_i,C_j)=\frac{(|P(C_i,C_j)|+1)\times r_{ij}}{d_{\mathrm{between}}(C_i,C_j)},\tag{14}$$

式中$|P(C_i,C_j)|$为小簇 C_i 和 C_j 之间交叉样本对数目。

通过式(14)可以看出，小簇间相似度与簇间交叉样本对数目和簇间密度关系值成正比，与簇间距离成反比。若簇间交叉样本对数目越多，簇间距离越近，簇间密度关系值越大，则簇间相似度越高，说明两个小簇同属一类的概率也就越大。当获得小簇间相似度矩阵后，按照最大相似度原则，将小簇合并。始终合并相似度最高的两个小簇，直到聚类数目达到期望的类别数后，停止合并，输出聚类结果。

为验证所提小簇合并策略的有效性，仍以 Lineblobs 数据集进行试验，试验结果如图 8 所示。图 8 为采用高斯核密度小簇合并策略在 Lineblobs 数据集上得到的最终聚类结果，可以发现该结果与图 2(a)中 Lineblobs 数据集原始分布完全相同，说明利用小簇合并策略在 Lineblobs 数据集上获得了完全正确聚类结果，体现了所提小簇合并策略有效性。

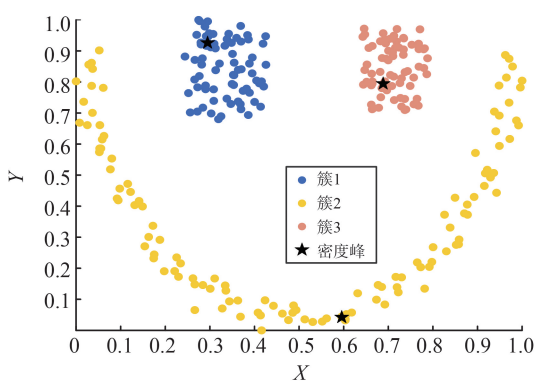


图 8　Lineblobs 数据集上聚类结果

Fig.8　Clustering result on Lineblobs

2.3　LSDPC 算法流程

LSDPC 算法通过计算样本局部密度,获得潜在密度峰以形成多个小簇,利用小簇合并策略分配样本。基于上述分析,给出 LSDPC 算法流程如下:

输入　数据集 D,调控因子 α;

输出　聚类结果 L;

步骤 1　数据归一化,获得样本之间的距离矩阵;

步骤 2　利用式(5~6)计算样本局部截断距离和截断近邻,利用式(7)计算样本局部密度;

步骤 3　利用式(3)获得每个样本相对距离,将样本密度和相对距离归一化,由式(4)计算每个样本决策值;

步骤 4　将决策值降序排列,按照样本决策值的差值关系选择潜在密度峰,将其余样本按照原始 DPC 算法分配策略进行聚类,形成多个小簇;

步骤 5　根据式(10~13)分别计算簇内平均密度、簇间距离、簇间交叉样本对和簇间密度关系值,由式(14)计算小簇间相似度;

步骤 6　将相似度降序排列,始终合并相似度最大的两个小簇,直到类总数达到期望的类数目,聚类停止;

步骤 7　将每个类簇赋予不同的标签,输出聚类结果 L。

2.4　LSDPC 算法时间复杂度

假设数据集的样本总数为 n,形成的小簇数目为 M,LSDPC 算法时间复杂度主要有:(1)获得样本间距离矩阵时间复杂度 $O(n^2)$;(2)计算样本密度时间复杂度 $O(n)$;(3)获得每个样本相对距离时间复杂度 $O(n)$;(4)选择潜在密度峰并形成小簇时间复杂度为 $O(n^2)$;(5)小簇合并阶段,通过计算簇内平均密度、簇间距离、簇间交叉样本对和簇间密度关系值来获得簇间相似度时间复杂度为 $O(M^2)$。由于 n 远大于 M,综上所述,LSDPC 算法总时间复杂度为 $O(n^2)$。

3　仿真试验及结果分析

3.1　试验设置

为验证 LSDPC 算法性能,将其与 DPC[13]、DPCV[17]、DPCSA[20]、FHC-LDP[23]和 DPC-CE[24]算法在 6 个复杂结构人工数据集[29]和 8 个 UCI 数据集[30]上进行对比试验。表 1 给出了所有数据集详细信息。

表 1　数据集详细信息

Table 1　Details of the datasets

数据集	样本数量/个	属性数量/个	类数量/个	类型
Lineblobs	266	2	3	人工
S2	5 000	2	15	人工
Donut2	1 000	2	2	人工
2d4c2	863	2	4	人工
Unbalance	6 500	2	8	人工
Square4	1 000	2	4	人工
Wine	178	13	3	UCI
Vote	435	16	2	UCI
Ecoli	336	7	8	UCI
Thyroid	215	5	3	UCI
Pima	768	8	2	UCI
Zoo	101	16	7	UCI
Breast	277	9	2	UCI
Landsat	2 000	36	6	UCI

LSDPC 算法参数 α 为[1,6],步长为 0.05,上述对比算法除了 DPCSA 和 DPC-CE 算法不需要参数调优外,其余算法均需参数调优。DPCV 算法参数 d_c 和 k 分别为[0.1,3]和[1,30],步长分别为 0.1 和 1;FHC-LDP 算法的参数 k 为[1,40],步长为 1;DPC 算法参数 d_c 为[0.05,2],步长为 0.01。算法试验环境为 Win11 64bit 操作系统、i7-12700H 处理器、16 G 内存和 MATLAB R2022b 软件。在所有试验中,算法的评价指标采用标准化互信息 N_{MI}、调整兰德系数 A_{RI} 和调整互信息 A_{MI} 这 3 个外部评价指标[31],各指标最优上限均为 1,指标值越接近 1 说明算法聚类性能越好。

3.2　试验结果及分析

表 2 给出了 6 种算法在 14 个数据集上试验结果。表 2 中,A_{RG} 为算法得到最优结果时参数取值,DPCV 算法涉及两个参数,"/"之前为参数 d_c,之后为参数 k;"-"表示算法无需调节参数;t 为算法运行时间,单位为 s。公平起见,将最佳参数应用于每个数据集,重复此过程 10 次取平均值得到算法在每个数据集上的运行时间。在表 2 中,将各个数据集上最优评价指标值和最短时间加粗显示。

观察表 2 中各算法在 6 个人工数据集上评价指标 N_{MI}、A_{RI} 和 A_{MI} 的结果,发现 LSDPC 算法在这些数据集上都获得了最优值。FHC-LDP 算法在 Lineblobs、Unbalance 和 2d4c2 数据集上获得了与 LSDPC 算法相同的聚类效果。DPCSA 算法在 Lineblobs 和 2d4c2 数据集上取得了与 LSDPC 算法相同的聚类结果。DPCV 算法在 Lineblobs 和

Unbalance 数据集上取得了和 LSDPC 算法一样的聚类结果。DPC 和 DPC-CE 算法在各数据集的聚类结果都比较差,说明 DPC 和 DPC-CE 算法无法处理结构复杂的数据集。一些算法在变密度数据集(Unbalance 数据集)和流形数据集(Lineblobs 数据集)上取得与 LSDPC 算法相同或相近的聚类结果,这些算法未充分考虑样本间相互关系,导致这些算法在不同类之间存在交叉样本数据集(S2、Square4 和 Donut2 数据集)上的聚类结果均明显劣于 LSDPC 算法。

表 2 各算法在所有数据集上试验结果

Table 2 Experimental results of each algorithm on all datasets

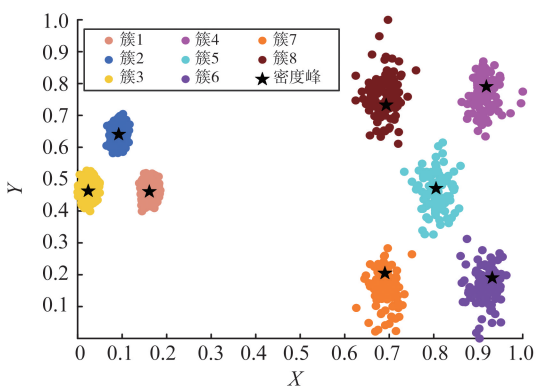
算法	S2					Lineblobs				
	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}
LSDPC	**0.947 5**	**0.939 9**	**0.947 1**	0.708	1.25	**1.000 0**	**1.000 0**	**1.000 0**	**0.006**	1.00
DPCSA	0.934 1	0.915 2	0.933 3	1.269	—	**1.000 0**	**1.000 0**	**1.000 0**	0.029	—
DPCV	0.942 6	0.931 0	0.941 4	74.470	0.10/30	**1.000 0**	**1.000 0**	**1.000 0**	0.029	2.30/25
FHC-LDP	0.944 6	0.935 5	0.944 2	**0.038**	36	**1.000 0**	**1.000 0**	**1.000 0**	0.011	18
DPC	0.943 6	0.934 3	0.943 2	0.962	0.990	0.703 4	0.648 2	0.694 7	0.013	0.36
DPC-CE	0.945 7	0.937 0	0.945 3	1.373×10^4	—	0.663 2	0.471 4	0.614 9	0.249	—
算法	Unbalance					2d4c2				
	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}
LSDPC	**1.000 0**	**1.000 0**	**1.000 0**	1.601	1.00	**0.993 0**	**0.995 9**	**0.992 8**	**0.034**	1.00
DPCSA	**1.000 0**	**1.000 0**	**1.000 0**	2.859	—	**0.993 0**	**0.995 9**	**0.992 8**	0.062	—
DPCV	0.897 0	0.862 3	0.897 0	163.234	0.10/5	0.958 3	0.916 0	0.919 0	0.420	2.20/7
FHC-LDP	**1.000 0**	**1.000 0**	**1.000 0**	**0.105**	30	**0.993 0**	**0.995 9**	**0.992 8**	**0.034**	14
DPC	0.897 2	0.858 2	0.897 2	1.718	0.29	0.900 3	0.893 2	0.824 5	0.044	1.30
DPC-CE	0.926 3	0.883 4	0.926 3	1.394×10^4	—	0.875 6	0.864 7	0.824 2	16.677	—
算法	Square4					Donut2				
	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}
LSDPC	**0.739 1**	**0.775 6**	**0.738 0**	0.041	1.60	**0.981 2**	**0.992 0**	**0.981 2**	0.038	1.60
DPCSA	0.694 0	0.713 3	0.689 6	0.071	—	0.966 4	0.984 0	0.966 3	0.064	—
DPCV	0.732 4	0.768 9	0.730 7	0.609	2.60/7	0.966 4	0.984 0	0.966 3	0.589	0.20/10
FHC-LDP	0.728 5	0.773 0	0.727 4	**0.009**	15	0.966 4	0.984 0	0.966 3	**0.009**	40
DPC	0.719 4	0.744 1	0.716 1	0.073	0.80	0.266 2	0.177 5	0.231 3	0.047	0.28
DPC-CE	0.706 8	0.739 3	0.704 6	26.586	—	0.146 0	0.032 1	0.095 8	22.841	—
算法	Landsat					Breast				
	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}
LSDPC	0.626 7	0.516 9	0.606 5	0.161	4.85	**0.096 2**	**0.162 2**	**0.069 6**	**0.007**	6.00
DPCSA	0.620 0	0.543 2	0.571 8	0.240	—	0.000 6	0.008 6	−0.002 6	0.023	—
DPCV	0.544 6	0.455 6	0.541 9	5.232	1.0/10	0.062 7	0.145 2	0.056 2	0.030	0.1/25
FHC-LDP	**0.658 7**	**0.649 0**	**0.647 8**	**0.067**	11	0.020 4	0.047 2	0.008 9	**0.005**	13
DPC	0.549 3	0.455 4	0.544 5	0.203	0.14	0.052 8	0.129 6	0.046 4	0.010	0.06
DPC-CE	0.545 4	0.442 0	0.540 9	224.021	—	0.008 6	0.042 4	0.004 4	0.025	—
算法	Zoo					Thyroid				
	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}
LSDPC	**0.871 7**	**0.894 6**	**0.823 2**	**0.003**	5.40	**0.486 5**	**0.520 6**	**0.396 1**	**0.005**	2.45
DPCSA	0.730 4	0.510 8	0.677 2	0.026	—	0.328 3	0.318 5	0.231 4	0.020	—
DPCV	0.733 2	0.727 5	0.642 4	0.008	2.1/23	0.484 2	0.445 9	0.349 6	0.019	0.1/20
FHC-LDP	0.837 0	0.846 5	0.783 1	0.005	12	0.393 2	0.370 1	0.363 3	0.006	4
DPC	0.492 5	0.213 9	0.400 8	0.006	0.05	0.323 4	0.159 0	0.288 4	0.008	0.75
DPC-CE	0.372 0	0.025 2	0.245 2	0.009	—	0.173 6	0.157 4	0.151 8	0.153	—

表 2(续)

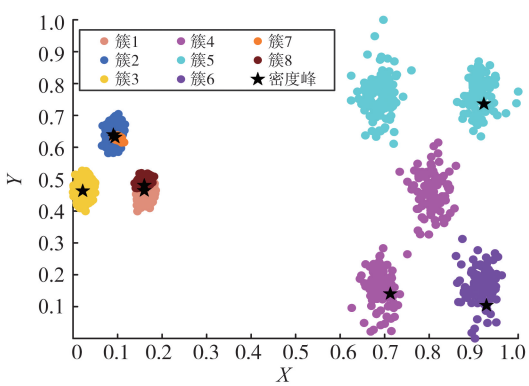
算法	Wine					Ecoli				
	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}
LSDPC	**0.795 5**	**0.802 5**	**0.789 8**	**0.004**	4.40	**0.655 5**	**0.732 9**	**0.623 2**	0.007	1.50
DPCSA	0.547 5	0.372 0	0.409 3	0.021	—	0.547 8	0.425 4	0.473 4	0.033	—
DPCV	0.710 4	0.672 4	0.706 5	0.014	1.8/8	0.640 3	0.686 2	0.538 3	0.045	0.1/7
FHC-LDP	0.743 5	0.726 9	0.739 1	0.011	20	0.651 8	0.697 4	0.544 1	**0.006**	38
DPC	0.617 0	0.606 8	0.610 0	0.007	0.65	0.589 3	0.436 5	0.569 3	0.017	0.19
DPC-CE	0.616 8	0.556 0	0.596 1	0.014	—	0.576 1	0.422 4	0.526 5	0.420	—

算法	Pima					Vote				
	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}	N_{MI}	A_{RI}	A_{MI}	t/s	A_{RG}
LSDPC	**0.074 5**	**0.100 4**	**0.050 9**	0.028	3.15	**0.531 4**	**0.613 5**	**0.521 8**	0.014	5.10
DPCSA	0.001 6	0.011 4	0.000 4	0.048	—	0.468 7	0.536 8	0.476 6	0.032	—
DPCV	0.023 3	0.058 4	0.021 8	0.318	0.4/4	0.505 9	0.564 1	0.495 8	0.086	1.5/2
FHC-LDP	0.005 2	0.014 3	0.001 7	**0.009**	21	0.464 4	0.536 7	0.455 1	**0.011**	25
DPC	0.010 4	0.034 1	0.006 7	0.036	0.18	0.469 4	0.536 7	0.459 8	0.021	0.23
DPC-CE	0.031 8	0.067 7	0.023 2	1.475	—	0.456 0	0.510 1	0.446 4	0.034	—

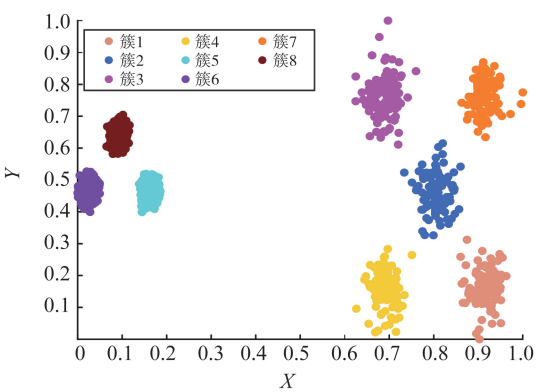
为直观展现聚类效果,图 9 给出了各个算法在 Unbalance 数据集上聚类效果图。图中不同颜色代表数据集不同类簇,黑色五角星为各个类簇的密度峰。Unbalance 数据集是一个典型变密度数据集,位于最左边 3 个簇的密度要明显高于右边 5 个簇。从图 9 可以发现,LSDPC 算法在各个类簇上准确识别出密度峰,获到了完全正确聚类结果。FHC-LDP 和 DPCSA 算法也对样本局部分布进行了考虑,同样取得了和 LSDPC 算法相同的结果。鉴于 DPCV 算法仍使用全局分布信息改进局部密度,该算法未在 Unbalance 上取得正确的密度峰,获得了较差的聚类结果。DPC 和 DPC-CE 算法利用原始截断距离定义密度,导致其在数据集左边单个高密度簇中产生了多个密度峰,将该高密度类簇中的部分样本错误分配,这两种算法在该数据集上的聚类结果都不够理想。基于上述分析,LSDPC 算法在复杂结构人工数据集中均具有较好聚类性能。



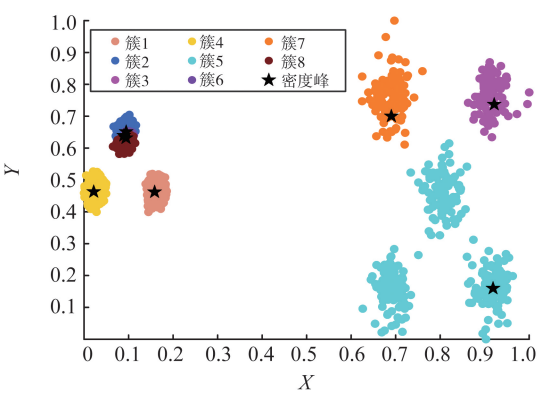
(a) LSDPC



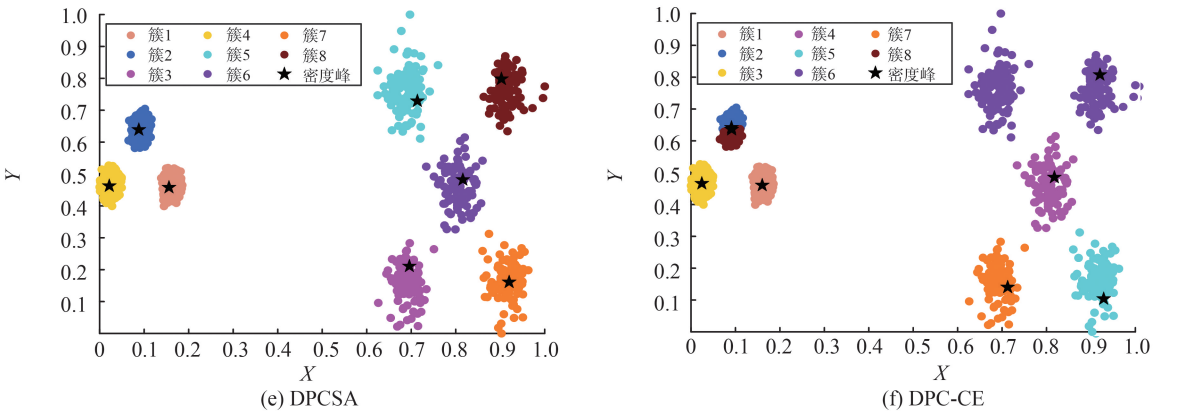
(b) DPC



(c) FHC-LDP



(d) DPCV



(e) DPCSA (f) DPC-CE

图 9 Unbalance 数据集上聚类结果

Fig.9 Clustering results obtained on Unbalance

观察表 2 中各个算法在 8 个 UCI 数据集上评价指标 N_{MI}、A_{RI} 和 A_{MI} 的结果,发现除 Landsat 数据集外,LSDPC 算法在其余 7 个数据集上均获得了最优评价指标值。LSDPC 算法在 Landsat 数据集上评价指标值略低于 FHC-LDP 算法在该数据集上取得的最优值,明显领先其他算法在该数据集上获得的评价指标值。在其余 7 个数据集上,LSDPC 算法取得了最优值,较对比算法而言,聚类性能优势明显。原因是 LSDPC 算法综合考虑了样本局部分布信息和类簇间相互关系,提高了聚类准确性。综上所述,LSDPC 算法在面对具有更高维度,数据分布也更为复杂的 UCI 数据集时,也具有较好聚类性能。在所有数据集上评价指标 N_{MI}、A_{RI} 和 A_{MI} 的平均值比 5 个对比算法平均提高 18.15%、28.99% 和 20.22%,比原始 DPC 算法提高 30.06%,47.15% 和 31.90%。

由表 2 可知 LSDPC 算法在各个数据集上运行时间均少于 DPCV、DPCSA、DPC 和 DPC-CE 算法,原因是 LSDPC 算法将样本划分到不同小簇中,通过处理小簇而不是每个样本提高了算法的效率。在样本数较多的数据集(S2、Unbalance 和 Landsat 数据集)上,得益于 FHC-LDP 算法较低的时间复杂度($O(n\log n)$),运行效率优于 LSDPC 算法;在部分样本数较少的数据集(Lineblobs、2d4c2、Zoo、Thyroid 和 Ecoli 数据集)上,LSDPC 算法的运行时间接近或领先 FHC-LDP 算法。LSDPC 和 FHC-LDP 算法在所有数据集上的平均运行时间分别为 0.189 7 s 和 0.023 2 s,前者运行时间略高于后者,原因是 LSDPC 算法需要计算距离矩阵,综合考虑小簇间相互关系定义簇间相似度,再基于该相似度进行聚类。正是因为这样的算法流程,使 LSDPC 算法比 FHC-LDP 算法花费了相对较多时间,这也使得 LSDPC 算法获得了相对较好聚类效果。

为了进一步验证所提算法有效性,利用秩均值对各算法性能进行检验。秩和检验是一种非参数统计检验方法[32],秩均值越高,该算法聚类性能就越好。表 3 列出了各算法在所有数据集上评价指标秩均值排名情况,最优秩均值加粗显示。

表 3 算法在所有数据集上的秩均值

Table 3 Average rank of the algorithm on all datasets

算法	N_{MI}	A_{RI}	A_{MI}
LSDPC	**5.68**	**5.61**	**5.68**
DPCSA	2.68	3.04	2.68
DPCV	3.54	3.82	3.54
FHC-LDP	4.14	4.25	4.21
DPC	2.50	2.46	2.79
DPC-CE	2.07	1.93	2.00

由表 3 可知,LSDPC 算法在各个评价指标上的秩均值明显领先于所对比 DPC 及其改进算法,秩均值在 N_{MI}、A_{RI} 和 A_{MI} 这 3 个指标上都获得了最好排名,进一步验证了 LSDPC 算法在各种类型数据集上聚类性能的优越性。

3.3 LSDPC 算法参数分析

LSDPC 算法所涉及的参数为调控因子 α,通过试验可知,调控因子的选择直接影响算法性能,本节在 0.5~8.0 之间对调控因子 α 进行分析。图 10 展示了在 Donut2、S2、Zoo 和 Unbalance 数据集上不同参数选择对算法性能的影响,图 10 中使用 N_{MI}、A_{RI} 和 A_{MI} 指标表示算法性能。

通过图 10 可以发现,当参数在 1~6 之间时,算法在各数据集上可以取得最优值,在该范围内算法对参数 α 不敏感。当 α 过大或者过小时,就会引起局部截断距离过大或过小,使得新定义的局部密度无法有效反映样本局部分布情况,导致 LSDPC 算法无法取得最优值。综上所述,在给定区间内,算法在各个数据集上能够获得最佳聚类结果,算法对参数 α 具有一定稳定性。

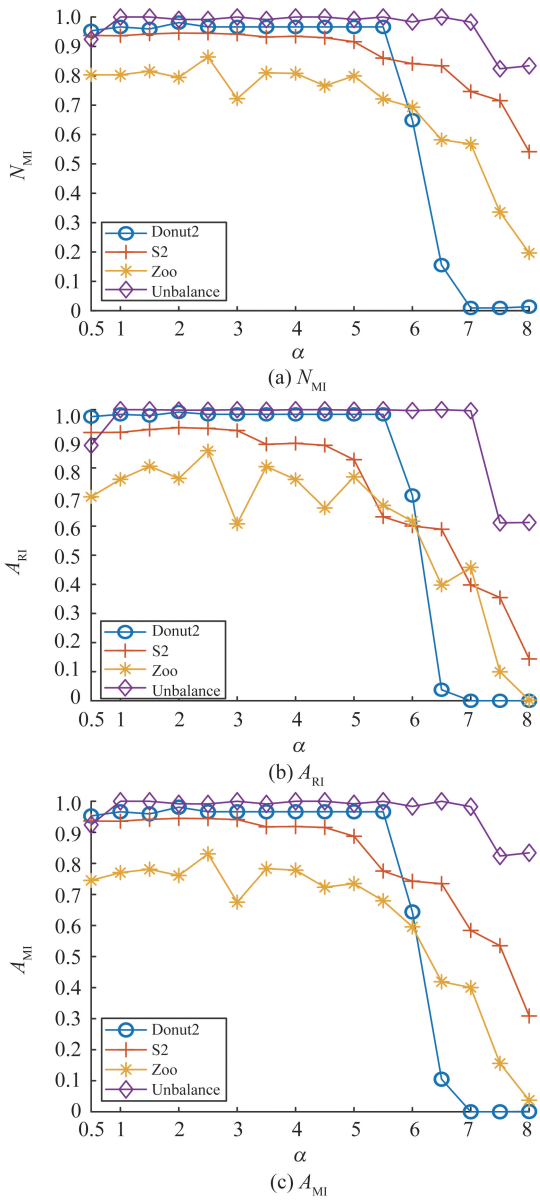


图 10　算法参数分析

Fig.10　Algorithm parameter analysis

4　结论

本研究提出了融合局部截断距离及小簇合并的密度峰值聚类算法。该算法基于样本所属区域的局部分布信息定义了样本局部截断距离和局部密度,统一了原始 DPC 算法密度定义方式;利用样本决策值之间差值关系选择潜在密度峰形成多个小簇,有利于准确刻画类簇结构;基于样本间相互关系定义了一种新的簇间相似度,利用该相似度进行小簇合并获得聚类结果,避免了“多米诺现象”发生。在复杂结构人工数据集和 UCI 数据集上进行试验,与 DPC 及其改进算法相比较,试验结果说明了所提 LSDPC 算法聚类性能优于上述比较算法,可以有效识别变密度数据集、流形数据集等复杂结构数据集。但所提 LSDPC 算法的聚类性能对调控因子 α 有一定依赖,算法时间复杂度相对于原始 DPC 算法没有明显优势,如何自适应选择最优调控因子和构造新的相似度度量方式来提高算法对大规模、高维数据处理效率将是下一步的研究重点。

参考文献:

[1] JING Wenbo, JIN Tian, XIANG Deliang. Fast superpixel-based clustering algorithm for SAR image segmentation[J]. IEEE Geoscience and Remote Sensing Letters, 2022, 19:1-5.

[2] 颜长春, 廖俊. 中国平台经济发展水平评价指标体系构建与测度[J]. 统计与决策, 2023, 39(11): 5-10.
YAN Changchun, LIAO Jun. Platform of China economic development level evaluation index system construction and measure[J]. Journal of Statistics and Decision, 2023, 33(11): 5-10.

[3] ZHAO Yuan, FANG Zhaoyu, LIN Cuixiang, et al. RFCell: a gene selection approach for scRNA-seq clustering based on permutation and Random Forest[J]. Frontiers in Genetics, 2021, 12: 665843.

[4] GAO Tengfei, CHEN Dan, TANG Yunbo, et al. Adaptive density peaks clustering: towards exploratory EEG analysis [J]. Knowledge-Based Systems, 2022, 240: 108123.

[5] LI Chuanwei, CHEN Hongmei, LI Tianrui, et al. A stable community detection approach for complex network based on density peak clustering and label propagation[J]. Applied Intelligence, 2022, 52(2): 1188-1208.

[6] GUAN X, TERADA Y. Sparse kernel *k*-means for high-dimensional data [J]. Pattern Recognition, 2023, 144: 109873.

[7] WANG W, YANG J, MUNTZ R. STING: a statistical information grid approach to spatial data mining[C]// Proceedings of 23rd International Conference on Very Large Data Bases. San Francisco, USA: ACM, 1997: 186-195.

[8] ANDREAS L, ERICH S. BETULA: fast clustering of large data with improved BIRCH CF-Trees [J]. Information Systems, 2022, 108: 101918.

[9] HUANG X G, MA T F, LIU C, et al. GriT-DBSCAN: a spatial clustering algorithm for very large databases[J]. Pattern Recognition, 2023, 142: 109658.

[10] FU Nan, NI Weiwei, HU Haibo, et al. Multidimensional grid-based clustering with local differential privacy[J]. Information Sciences, 2023, 623: 402-420.

[11] SUN Mingchen, YANG Mengduo, LI Yingji, et al.

Structural-aware motif-based prompt tuning for graph clustering[J]. Information Sciences, 2023, 649: 119643.

[12] WU Chengmao, YU Dongxue. Generalized possibilistic c-means clustering with double weighting exponents[J]. Information Sciences, 2023, 645: 119283.

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

[14] WANG Y Z, QIAN J X, HASSAN M, et al. Density peak clustering algorithms: a review on the decade 2014—2023 [J]. Expert Systems with Applications, 2024, 238: 121860.

[15] WEI Xiuxiu, PENG Maosong, HUANG Huajuan, et al. An overview on density peaks clustering [J]. Neurocomputing, 2023, 554: 126633.

[16] 陈叶旺, 申莲莲, 钟才明,等. 密度峰值聚类算法综述[J]. 计算机研究与发展, 2020, 57(2): 378-394.
CHEN Yewang, SHEN Lianlian, ZHONG Caiming, et al. Survey on density peak clustering algorithm [J]. Journal of Computer Research and Development, 2020, 57(2): 378-394.

[17] DING Shifei, DU Wei, LI Chao, et al. Density peaks clustering algorithm based on improved similarity and allocation strategy[J]. International Journal of Machine Learning and Cybernetics, 2023, 14(4): 1527-1542.

[18] ZHOU Zhou, SI Gangquan, SUN Haodong, et al. A robust clustering algorithm based on the identification of core points and KNN kernel density estimation [J]. Expert Systems with Applications, 2022, 195: 116573.

[19] YAN Huan, WANG Mingzhao, XIE Juanying. ANN-DPC: density peak clustering by finding the adaptive nearest neighbors[J]. Knowledge-Based Systems, 2024, 294: 111748.

[20] YU Donghua, LIU Guojun, GUO Maozu, et al. Density peaks clustering based on weighted local density sequence and nearest neighbor assignment [J]. IEEE Access, 2019, 7: 34301-34317.

[21] ZHAO J, WANG G, PAN J S, et al. Density peaks clustering algorithm based on fuzzy and weighted shared neighbor for uneven density datasets [J]. Pattern Recognition, 2023, 139: 109406.

[22] 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 Jia, YAO Zhanfeng, LÜ Li, et al. Density peaks clustering based on mutual neighbor degree[J]. Control and Decision, 2021, 36(3): 543-552.

[23] GUAN Junyi, LI Sheng, HE Xiongxiong, et al. Fast hierarchical clustering of local density peaks via an association degree transfer method [J]. Neurocomputing, 2021, 445: 401-418.

[24] GUO Wenjie, WANG Wenhai, ZHAO Shunping, et al. Density peak clustering with connectivity estimation[J]. Knowledge-Based Systems, 2022, 243: 108501.

[25] QIAO Kaikai, CHEN Jiawei, DUAN Shukai. Self-adaptive two-stage density clustering method with fuzzy connectivity [J]. Applied Soft Computing, 2024, 154: 111355.

[26] 赵嘉, 马清, 肖人彬,等. 面向流形数据的共享近邻密度峰值聚类算法[J]. 智能系统学报, 2023, 18(4): 719-730.
ZHAO Jia, MA Qing, XIAO Renbin, et al. Shared neighbor density peak clustering algorithm for manifold data [J]. CAAI Transactions on Intelligent Systems, 2023, 18(4): 719-730.

[27] CHENG Dongdong, ZHU Qingsheng, HUANG Jinlong, et al. Clustering with local density peaks-based minimum spanning tree[J]. IEEE Transactions on Knowledge and Data Engineering, 2021, 33(2): 374-387.

[28] QIU Teng, LI Yongjie. Fast LDP-MST: an efficient density-peak-based clustering method for large-size datasets[J]. IEEE Transactions on Knowledge and Data Engineering, 2023, 35(5): 4767-4780.

[29] MILAAN P. Clustering datasets[EB/OL]. (2022-12-10) [2023-10-11]. https://github.com/milaan9/Clustering-Datasets

[30] BLAKE C L, MERZ C J. UCI repository of machine database [EB/OL]. (2023-09-29) [2023-10-15]. https://archive.ics.uci.edu/datasets

[31] VINH N, EPPS J, BAILEY J. Information theoretic measures for clustering comparison: variants, properties, normalization and correction for chance[J]. Journal of Machine Learning Research, 2010, 11: 2837-2854.

[32] DONALD W Z, BRUNO D Z. Relative power of the Wilcoxon Test, the Friedman Test, and Repeated-Measures ANOVA on Ranks [J]. The Journal of Experimental Education, 1993, 62(1): 75-86.

(编辑:陈燕)