计算机科学与探索
Journal of Frontiers of Computer Science and Technology
1673-9418/2022/16(08)-1910-13
doi: 10.3778/j.issn.1673-9418.2111138

去中心化加权簇归并的密度峰值聚类算法

赵力衡[1+],王　建[1,2],陈虹君[1]
1. 成都锦城学院 电子信息学院,成都 611731
2. 四川大学 计算机学院,成都 610041
+ 通信作者 E-mail: 1503233800@qq.com

摘　要:快速搜索和寻找密度峰值聚类算法(DPC)是近年来提出的一种基于密度的聚类算法,具有原理简单、无需迭代并能实现任意形状聚类的优点。但该算法仍存在一些缺陷:围绕聚类中心点聚类,使聚类结果受中心点影响显著,且聚类中心点数量仍需人为指定;截断距离仅考虑了数据的分布密度,忽略了数据的内部特征;聚类过程中若有样本存在分配错误,会导致其后续样本聚类出现跟随错误。针对上述问题,尝试提出一种去中心化加权簇归并的密度峰值聚类算法(DCM-DPC)。该算法引入权重系数重新定义了局部密度,并由此划分出位于不同局部高密度区域的核心样本组,用于取代聚类中心点成为聚类的依据。最后将剩余样本按其近邻样本所在类簇的众数,或分配到最高耦合的核心样本组代表的类簇中或标注为离散点以完成聚类。在人工和UCI数据集上的实验结果表明,提出算法的聚类效果优于对比算法,对相互纠缠的类簇的边界样本划分也更加精确。

关键词:密度峰值;聚类;去中心点;邻域;簇归并
文献标志码:A　　**中图分类号:**TP301

Density-Peak Clustering Algorithm on Decentralized and Weighted Clusters Merging

ZHAO Liheng[1+], WANG Jian[1,2], CHEN Hongjun[1]
1. Department of Electronic Information Engineering, Chengdu Jincheng College, Chengdu 611731, China
2. School of Computer, Sichuan University, Chengdu 610041, China

Abstract: The clustering by fast search and find of density peaks (DPC) is a density-based clustering algorithm proposed in recent years, which has the advantages of simple principle, no iteration and clustering of arbitrary shape. However, the algorithm still has some defects: clustering around clustering centers makes the clustering results significantly affected by central points, and the number of clustering centers needs to be manually specified; the cutoff distance considers the distribution density of the data but ignores the internal features; if there is a sample allocation error in the clustering process, the subsequent sample clustering may amplify the error. To solve the above problems, this paper proposes a density-peak clustering algorithm on decentralized and weighted clusters merging (DCM-DPC). This algorithm introduces the weight to redefine the local density, dividing core sample groups located in different local high density regions to replace cluster centers as the cluster basis. Finally, the remaining samples are assigned to the highest coupled core sample groups or labeled as discrete points by their near neighbor samples. Experiments on artificial and UCI datasets show that the clustering performance of the proposed algorithm

基金项目:教育部协同育人项目(201902005069);四川省科技厅重点研发项目(22ZDYF0724)。
This work was supported by the Collaborative Education Project of Ministry of Education of China (201902005069), and the Key Research and Development Project of Sichuan Provincial Science and Technology Department (22ZDYF0724).
收稿日期:2021-11-30　　**修回日期:**2022-03-24

outperforms the contrast algorithms, and the boundary samples partition of the entangled clusters is more accurate.
Key words: density peaks; clustering; decentralized; neighborhood; clusters merging

聚类算法是模式识别和数据挖掘领域中的一类常见的无监督算法[1]。算法通过某种相似性计算将一组数据对象按照其自身的特征划分到不同的类簇中，并使同一簇内的对象尽可能相似，不同簇之间的对象尽可能不相似[2]。现有聚类算法模型丰富，大致可以分为[3-6]层次聚类、划分聚类、密度聚类、网格聚类、图论聚类、代表点聚类和模型聚类七种类型，被广泛应用于生物[7]、能源[8]、交通[9]、图像处理[10]、流形数据处理[11]等领域。

聚类算法自应用以来，不断有学者提出新的聚类算法，Rodriguez等人[12]于2014年在 *Science* 上提出了快速搜索和寻找密度峰值聚类算法（clustering by fast search and find of density peaks，DPC）。该算法是一种基于密度的聚类算法，依据样本的分布密度能无需迭代地快速找出任意形状数据集中的密度峰值样本，以之作为聚类中心，从而能够较高效地得到高精确的聚类结果。DPC算法因此受到了广泛的关注，但该算法依旧存在着一些缺陷：(1)围绕聚类中心点进行聚类，若聚类中心点选取不当会对聚类结果造成显著影响；(2)DPC算法需要人工指定聚类中心，使算法的自动化程度受到影响；(3)截断距离只考虑了数据的分布密度，忽略了数据的内部特征，影响了聚类结果的稳定；(4)DPC在聚类过程中若存在样本分配错误，则错误样本邻近密度更低的样本将可能随之被分配到错误的类簇中，从而放大错误。

为解决这些问题，近年来众多学者对DPC算法进行了改进。Xie等人[13]提出了模糊加权K最近邻分配点的密度峰值聚类算法（robust clustering by detecting density peaks and assigning points based on fuzzy weighted K-nearest neighbors，FKNN-DPC）。该算法基于K近邻设计了一种独立于数据集规模且与截断距离无关的局部密度计算方式，将数据集划分成核心样本和离群样本，再采用新的K近邻策略完成对非聚类中心点的分配，有效缓解了DPC算法的跟随错误。Seyed等人[14]提出了基于动态图的密度峰值聚类标签传播算法（dynamic graph-based label propagation for density peaks clustering，DPC-DLP）。算法根据重新定义的K近邻密度确定出聚类中心点，然后将类簇中心点和其相邻中心点形成一个KNN图，最后采用图的标签传播方法分配剩余样本，更适用于图像聚类。丁世飞等人[15]提出了一种基于不相似性度量优化的密度峰值聚类算法（optimized density peaks clustering algorithm based on dissimilarity measure，DDPC）。算法通过基于块的不相似性度量实现样本间的相似度计算，从而避免了小样本数据集上截断距离对聚类结果的影响，提高了在高维度数据集上的聚类效果。Liu等人[16]提出了基于共享近邻的密度峰值聚类算法（shared-nearest-neighbor-based clustering by fast search and find of density peaks，SNN-DPC）。算法通过计算样本之间共享的近邻点个数，确定样本之间的相似度，避免了非聚类中心点分配时的跟随错误。王大刚等人[17]提出了基于二阶 k 近邻的密度峰值聚类算法（density peaks clustering algorithm based on second-order k neighbors，SODPC）。算法通过引入样本的二阶 k 近邻计算直接密度和间接密度，避免了截断距离带来的影响。

本文所知文献对DPC算法的改进主要着眼于聚类中心点的选取[18]、避免分配跟随错误[13]及效率[19]等方面，没有采用无中心点聚类的优化算法。尝试提出一种去中心化加权簇归并的密度峰值聚类算法（density-peak clustering algorithm on decentralized and weighted clusters merging，DCM-DPC）。算法在聚类过程中取消了聚类中心点的概念，认为位于彼此邻域内的局部高密度样本属于同一类簇，采用加权 ε 近邻思想，重新定义了样本邻域半径，从而划分出位于不同区域的局部高密度样本组，并在寻找样本组的过程中归并存在邻域重叠的区域，形成归并的核心样本组，最后将剩余样本按其近邻样本的众数归属到某个核心样本组中完成聚类。实验结果表明，DCM-DPC算法有效避免了由聚类中心点和截断距离带来的误差，并在聚类效果上有明显的提高。

1 DPC算法

DPC算法基于样本密度实现对数据集的聚类，算法假设类簇中心具有以下两个特征：(1)聚类中心点的局部密度高于周围样本的局部密度；(2)聚类中心点之间的距离相对较远。对于给定的数据集 $X=\{x_1,x_2,\cdots,x_n\}$，设每个元素的维度为 m。DPC算法定义样本 x_i 的局部密度 ρ_i 为：

$$\rho_i = \sum_{i \neq j} \chi(d_{ij} - d_c) \tag{1}$$

其中，d_{ij} 表示样本 x_i 与样本 x_j 之间的距离。d_c 为截断距离，定义为 X 中任意两个样本之间的距离按升序排列后位于用户指定位置的值。对于函数 $\chi(t)$ 有：

$$\chi(t) = \begin{cases} 1, x < 0 \\ 0, x \geqslant 0 \end{cases} \tag{2}$$

当数据集规模较小时，DPC采用高斯核函数[18]描述局部密度：

$$\rho_i = \sum_{j \neq i} \mathrm{e}^{-\left(\frac{d_{ij}}{d_c}\right)^2} \tag{3}$$

相对距离 δ_i 表示样本 x_i 距离局部密度比它高且离它最近的样本的距离，当 x_i 不是最大密度样本点时 δ_i 为：

$$\delta_i = \min_{j:\rho_j > \rho_i} (d_{ij}) \tag{4}$$

当 x_i 是最大密度样本点时 δ_i 为：

$$\delta_i = \max_j (d_{ij}) \tag{5}$$

DPC算法使用局部密度 ρ_i 和相对距离 δ_i 绘制出决策图，并选取 γ_i 最大的若干个样本作为聚类中心，聚类中心个数由用户指定：

$$\gamma_i = \rho_i \times \delta_i \tag{6}$$

以这些密度峰值点作为聚类中心，剩余的非聚类中心样本被分配给局部密度更高且距离最近的样本所在类簇，从而完成聚类。

DPC算法在多数时候能获得不错的聚类结果，但尚存在一些不足：

(1)围绕聚类中心点进行聚类，即首先找出聚类中心点，然后非聚类中心点依据聚类中心点进行分配，从而完成聚类。截断距离是影响聚类中心选取的重要因素，图1(a)和图1(b)分别是flame数据集在截断距离取5%和2%时的聚类结果，图中十字符号为聚类中心。可以看出，当截断距离不同时选取的距离中心不相同，聚类结果也出现显著差异。聚类过程中，不同截断距离除了影响聚类中心的选取外，还会引起局部密度等计算的变化，同样会影响聚类结果。因此，为消除聚类中心因素外其他因素对聚类结果的影响，图1(c)中将聚类效果优秀的截断距离采用5%的聚类中心替换为截断距离采用2%的聚类中心，替换后聚类结果中大部分样本被识别为离散点，聚类效果极差。可见聚类中心的选择可能显著影响聚类效果。

(2)通过决策图选取聚类中心，但聚类中心个数仍需人工指定，使算法的自动化程度受到影响。

(3)截断距离由用户主观选择，只体现了数据的分布密度，没有体现数据的内部特征，因此截断距离的改变容易使聚类结果变得不稳定。

(4)非聚类中心样本被分配给邻域密度大于该样本且距离其最近的样本所属的类簇。若一个样本分配错误，则该样本邻域内其他密度更小的样本就可能跟随该样本被分配到错误的类簇，形成“多米诺”效应，导致聚类结果不理想。

2 DCM-DPC算法

针对上述不足，本文尝试提出一种基于去中心化加权簇归并的密度峰值算法(DCM-DPC)，从消除聚类中心、簇归并和非核心样本分配策略三方面对DPC算法进行改进。

2.1 去中心化的加权核心样本组策略

根据图1的分析可以发现，聚类中心点的质量很重要，甚至能显著影响聚类效果，因此找出合适的聚类中心是现有密度峰值聚类算法的关键。从聚类算法的本质看，聚类是将相似的样本划分在一起，而不是将样本围绕某个中心点划分在一起，因此聚类中

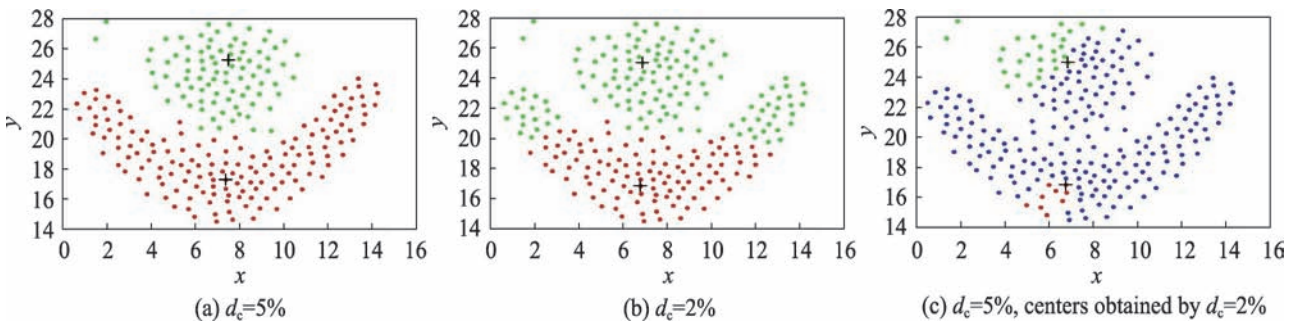

图1 不同聚类中心点的聚类效果对比图

Fig.1 Clustering effect contrast diagram of different clustering centers

心并不是必须的，若能识别出相似的样本，就能完成聚类。

本文所知的DPC改进算法[13-29]文献均依赖于聚类中心点进行聚类，并没有在消除聚类中心方向进行优化。尝试提出一种新的去中心化聚类的核心样本组策略取代聚类中心点作为样本划分依据。核心样本组指具有较高局部密度且位于同一较高密度区域样本的集合，采用基于 ε 近邻思想[11]的加权邻域半径来度量局部密度。ε 近邻思想目标是找出加权邻域半径内的所有样本数量。

DPC算法使用截断距离作为邻域半径，以截断距离内的样本数量作为局部密度。由于截断距离是人为主观选择，难以准确反映数据的分布特征，为此本文给出了新的局部密度及相关定义：

定义1（权重系数）设定权重系数如下：

$$\lambda = \frac{p_r}{\dfrac{\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^4}{\left(\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2\right)^2}-3} \tag{7}$$

式中，$\bar{x}$ 表示数据集 X 的均值，n 为样本数。

定义2（加权邻域半径 ε）设定加权邻域半径 ε 如下：

$$\varepsilon = \lambda \times \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}}{n^2} \tag{8}$$

式中，d_{ij} 表示样本 x_i 与 x_j 之间的距离。

加权邻域半径公式兼顾了数据的分布密度和内部特征，其中 $\dfrac{\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}}{n^2}$ 表示样本间距离均值。该值常用于描述数据平均水平的度量，能有效反映出数据的分布密度，但难以反映数据内部的结构差异，为此引入权重系数 λ。权重系数包括两部分：

（1）统计学中的样本峰度系数 $\dfrac{\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^4}{\left(\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2\right)^2}-3$，该系数用于描述样本局部密度之间的差异，其取值范围不低于1且不高于样本数。当样本间距离均值相等时，密度分布差异大的数据集中类簇的聚合程度显然比密度分布差异小的数据集更强，因此该系数能有效收敛邻域半径，反映出数据分布的不均匀性，从而提升聚类效果，且该系数对分布越不均匀的数据集的影响越显著。

（2）修正系数 p_r，数据集中的离散点对样本间距离均值的影响明显，容易导致邻域半径过大而失真，峰度系数对此修正不足，因此引入该系数用于修正邻域半径范围。

定义3（局部密度）局部密度定义如下：

$$\rho_i = \sum_{i\neq j} \chi(d_{ij}-\varepsilon) \tag{9}$$

式中，ε 为式（8）中表示的加权邻域半径。

以加权邻域半径内的样本数量作为局部密度，同时考虑到了数据的密度和内部结构的差异，能有效描述样本的分布状况，从而提升聚类效果。

算法依据局部密度将样本划分为核心样本、非核心样本和离散样本。

定义4（核心样本、非核心样本及离散样本）核心样本 c_i 指在加权邻域半径 ε 内的局部密度高于指定阈值 m_c 的数据点。非核心样本 b_i 指 ε 内密度不高于指定阈值 m_c 的数据点。离散样本 s_i 指 ε 内不存在可以归属于任意簇的样本的数据点，如式（10）所示：

$$\begin{cases} c_i = \{x_j \in X | \rho_i > m_c\} \\ b_i = \{x_j \in X | 0 < \rho_i \leqslant m_c\} \\ s_i = \{x_j \in \varepsilon_i | \forall x_j \notin A_k\} \end{cases} \tag{10}$$

其中，ε_i 表示 x_i 邻域半径内的样本，A_k 表示任意类簇。

本文算法以近邻样本之间共享的样本数来度量样本之间相似度。样本划分依据是，核心样本的近邻样本较多，因此容易判断与近邻样本之间的相似性，从而与相似样本组成类簇，并可作为聚类的依据。非核心样本通常位于较低局部密度的区域，由于近邻样本较少，不容易判断该样本与近邻样本的相似性，若作为聚类依据，容易发生漂移。若样本无近邻点或虽有近邻点但这些近邻点都不属于任何类簇，则该样本同样不能归属于任一类簇，因此需要被标注为离散样本。三者的关系是，核心样本集与非核心样本集互斥互补，离散样本集则是非核心样本集的子集。

DPC算法认为聚类中心点的局部密度在其周围样本中最高，可推断聚类中心点位于局部密度较高的区域，且其近邻存在局部密度较高的其他核心样本。如图2所示，若样本1是DPC算法的聚类中心点，在邻域半径 ε 内密度最高，样本3是样本1邻域内一个局部密度较高的核心样本，显然两者有较高的相似性。DPC算法认为聚类核心彼此距离较远，可以推断样本3附近不存在其他聚类核心，可知样本

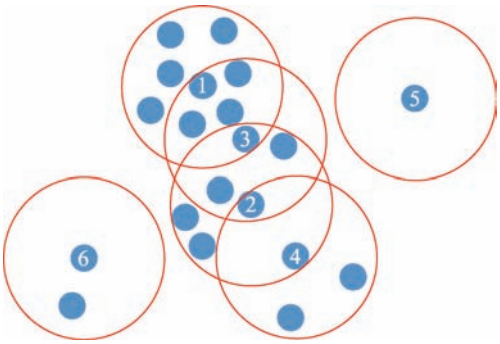

图2 数据分布示意图

Fig.2 Schematic diagram of data distribution

3归属于样本1所在的类簇。同理,假设样本2是样本3邻域内另一个核心样本,则样本2与样本3也具有较高的相似性,且同属于样本1所在类别。

可以发现,位于聚类中心点邻域半径内的核心样本和位于这些核心样本邻域半径内的其他核心样本同属于该聚类中心所在的类簇。样本4是样本1邻域内的非核心样本,近邻点较少因此与其近邻点相似度都不高,难以确定是否属于同一类别。显然,非核心样本的近邻样本中属于某个类簇的样本越多,该非核心样本就与该类簇越相似。因此本文算法以非核心样本的近邻点所属类簇的众数确定其归属。离散样本5、6因没有可以归属于任意类簇的近邻点,所以不属于任何类簇。可见,当聚类中心确定时,属于该类簇的核心样本成员,即核心样本组,亦就可以确定了,DPC算法的聚类中心点就是核心样本组中密度最高的点。核心样本组的寻找可以从任意核心样本开始,找出其近邻核心点,进而扩散到整个数据集,从而实现无中心点的聚类。

图3(a)展示了Aggregation数据集的样本分布图,由7个相邻且不同形状的类簇构成,分别以不同的颜色表示。图3(b)是该数据集不同类簇的核心样本和非核心样本的分布图,不同类簇的核心样本颜色与图3(a)中相同类簇的颜色相同,非核心样本则以其他颜色表示。可以发现,每个类簇的核心样本都集中在类簇中间密度较高的区域,非核心样本则围绕在核心样本组的周围局部密度较低的区域,且当类簇密度较高时,该类簇的核心样本也较多,反之则偏少。可见,由核心样本构成的核心样本组在反映密度峰值的意义上与DPC算法的聚类中心是一致的。DPC算法依据聚类中心聚类时,若样本密度差异较大,同一类簇中可能找到多个密度峰值[29],使聚类结果不理想,而核心样本组则会将这些密度峰值划分到同一核心样本组中,从而避免该现象。因此核心样本组不但能够成为聚类的依据,而且聚类效果优于DPC算法使用的聚类中心。

2.2 簇归并策略

在给定数据集 X 中,互为近邻的核心样本构成代表一个簇的核心样本组。识别核心样本的步骤中,只有核心样本会记录到代表类簇的核心样本组中,此时核心样本组等价于类簇。由于样本的顺序通常是未经排序的,当顺序遍历数据集寻找核心样本时,识别出的核心样本通常也是无序的,因此聚类过程中由核心样本组构成的类簇是变化的。由上节分析可知,互为近邻的核心样本属于同一类簇,因此当聚类过程中发现不同的核心样本组中存在相互近邻的样本时,表明这些样本组中的元素应属于同一类簇,需要将这些核心样本组归并成一个。

如图4所示,核心样本按标号顺序被识别出。核心样本3被识别时由于没有位于样本1的邻域半径内,样本1与样本3此时应分别属于不同的簇(核心样本组)。当同时位于二者邻域半径内的样本5被识别出时,可以发现三个样本是相似的,样本1和样本3所在的簇是相似簇,需要以样本5为介质进行归并。

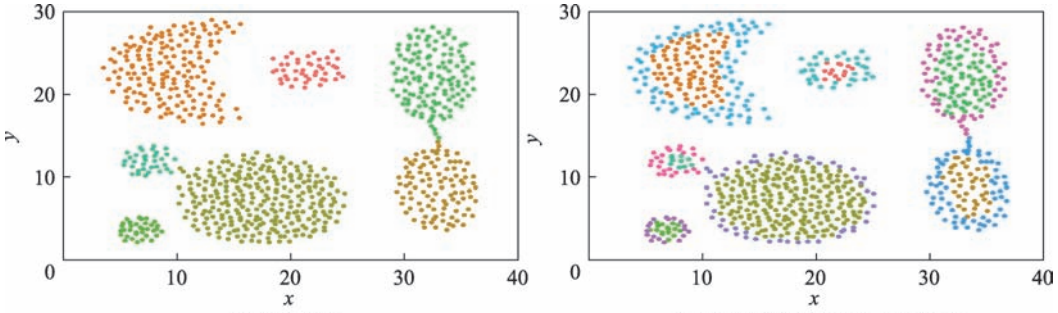

图3 Aggregation数据分布图

Fig.3 Distribution map of Aggregation dataset

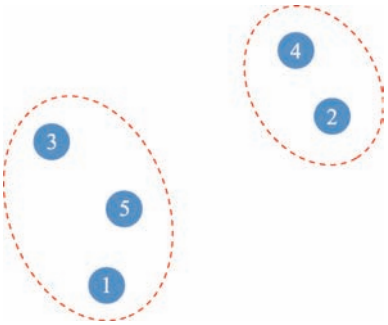

图4 核心样本组归并示意图

Fig.4 Schematic diagram of core sample groups

若不同核心样本所在簇不相似，则不进行归并，如样本1、3、5和样本2、4所在的簇。

定义5（类簇相似度）类簇相似度定义如下：

$$SIM(A_m,A_n)=\sum(c_i|c_i\in C_m\cap c_i\in C_n) \tag{11}$$

式中，C_m 和 C_n 分别表示类簇 A_m 和 A_n 的核心样本组，c_i 是 C_m 和 C_n 共享的核心样本。当 $SIM(A_m,A_n)\geqslant 1$ 时，类簇 A_m 和 A_n 相似。

当识别出所有核心样本后，没有归并的簇就组成了全部核心样本组。

2.3 非核心样本归属判定策略

样本与近邻点是相似的，越多近邻点属于同一类簇，表示样本与该类簇越相似，因此非核心样本 b_i 的归属采用近邻点所属类簇的众数来决定，包含 b_i 近邻点数 p_i 最多的类簇即为 b_i 最相似的类簇：

$$p_i=\max(N_\varepsilon(b_i)\in A_j) \tag{12}$$

其中，$N_\varepsilon(b_i)\in A_j$ 表示样本 b_i 属于类簇 A_j 的近邻点数量。当 $p_i=0$ 时，表示 b_i 为离散点。

DPC算法中非聚类中心样本 x_i 单纯依赖于距离最近且局部密度更高的样本 x_j，若 x_j 分配错误，x_i 会跟随分配错误，容错率很低。本文算法使非核心样本的分配由多个近邻点共同决定，大幅提高了样本划分的容错率，因此能有效避免跟随错误。

特别是当数据集中出现类簇纠缠时，边界样本更容易出现距离其他类簇中有更高局部密度的样本更近的现象，因此本文算法相对于DPC算法能更准确地识别出边界样本的所属类簇，使边界样本的分配更加精确可靠。

2.4 算法步骤

为消除样本属性之间量纲不一致带来的影响，本文将在计算前对数据进行归一化处理，将原始属性值通过线性变换映射到[0,1]区间。

定义6（数据归一化）样本 x_i 的属性 j 归一化定义如下：

$$x'_{ij}=\frac{x_{ij}-\min(x_{ij})}{\max(x_{ij})-\min(x_{ij})} \tag{13}$$

式中，$\max(x_{ij})$ 为样本 x_i 的属性 j 的最大值，$\min(x_{ij})$ 为样本 x_i 的属性 j 的最小值。

算法步骤如下：

输入：数据集 $X=\{x_1,x_2,\cdots,x_n\}$；核心对象邻域密度阈值 m_c；邻域半径修正权值 p_r。

步骤1 根据式(13)对数据归一化。

步骤2 根据式(8)计算加权邻域半径 ε。

步骤3 根据式(9)计算样本邻域密度 ρ_i，然后根据式(9)划分样本：

步骤3.1 将 $\rho_i>m_c$ 的样本录入其近邻核心样本所在核心样本组，若样本的近邻核心样本还未被识别或无近邻核心样本，则该样本录入新核心样本组；

步骤3.2 将 $\rho_i\leqslant m_c$ 的样本录入非核心对象队列中；

步骤3.3 每当识别出一个核心样本时，检查该样本是否为核心样本组的共享样本，如果是则合并相似类簇。

步骤4 完成核心样本识别后，对非核心样本按其近邻点所属类簇的众数，降序归入最相似的类簇中。

步骤5 标识非核心样本队列中剩余没有近邻点的样本为离散点。

输出：聚类结果集 R。

2.5 算法复杂度分析

对于样本规模为 n 的数据集，DPC算法的时间复杂度主要来自计算任意两个样本间的距离、计算所有样本的局部密度以及计算每对样本之间的相对距离。每部分的时间复杂度均为 $O(n^2)$，因此DPC算法的总时间复杂度为 $O(n^2)$。

本文DCM-DPC算法的时间复杂度主要来源于：(1)计算数据集加权邻域半径 ε 的时间复杂度 $O(n^2)$。(2)计算每个样本的局部密度的时间复杂度 $O(n^2)$。(3)簇归并的时间复杂度 $O(c^2)$，其中 c 为核心样本个数，c 小于样本个数 n，因此 $O(c^2)<O(n^2)$。(4)划分非核心样本并标注离散点的时间复杂度 $b<n\,O(kb^2)$，其中 k 为样本的近邻点个数，$k\ll n$，相比 n 来说可以忽略不记；b 为非核心样本个数，$b<n$，且 $c+b=n$，有 $O(kb^2)\approx O(b^2)<O(n^2)$，因此本文算法总时间复杂度为 $O(n^2)$，与DPC算法的时间复杂度相同。

3 实验结果与分析

3.1 实验数据集与评估指标

为验证DCM-DPC算法的有效性,本文采用人工数据集与UCI数据集进行测试和评估。为使测试数据多样化,选取的数据集在样本数量、属性数和类簇数跨度较大,这些数据集皆广泛地应用于聚类算法有效性的测试。数据集具体属性如表1和表2所示。

表1 人工数据集

Table 1 Artificial datasets

数据集	样本数	属性数	类簇数
Aggregation	788	2	7
D31	3 100	2	31
Flame	240	2	2
Jain	373	2	2
Spiral	312	2	3
R15	600	2	15

表2 UCI数据集

Table 2 UCI datasets

数据集	样本数	属性数	类簇数
Iris	150	4	3
Soybean(Small)	35	47	4
Statlog(Heart)	270	13	2

在以上数据集上选择DPC[12]、FKNN-DPC[13]、SNN-DPC[16]、DBSCAN[30]和*K*-means++[31]算法与本文DCM-DPC算法进行比较。其中,DPC和SNN-DPC算法使用的是作者公开的源代码,FKNN-DPC、DBSCAN和*K*-means++算法参照原文献使用Python3.8实现。本文依据参考文献对各算法的参数均进行了调优,以保证各算法的聚类效果。*K*-means++算法因初始聚类中心的选取具有随机性会影响聚类结果,表3和表4中采用100次聚类结果的均值。

评估指标采用调整互信息(adjusted mutual information,AMI)[32]、调整兰德系数(adjusted Rand index,ARI)[32]和FMI指数(Fowlkes Mallows index,FMI)[33]。其中,AMI和FMI取值范围为[0,1],ARI取值范围为[−1,1],三者均是越接近1,表明聚类效果越优。

3.2 实验结果分析

表3展示了6种算法在UCI数据集上的聚类结果,其中加粗字体表示较优的实验结果。实验结果显示,DPC和FKNN-DPC算法在属性数较多的数据集Soybean上和Statlog上聚类效果较差,但在属性较少的数据集Iris上相对于SNN-DPC和DBSCAN算法取得了显著的优势;SNN-DPC算法在Iris和Soybean(Small)数据集上的聚类指标相对较差,但在Statlog(Heart)上取得了较好的聚类结果;*K*-means++算法聚类效果正好与SNN-DPC算法相反;DBSCAN算法在3个数据集上的聚类结果都不太理想;DCM-DPC算法在Iris数据集上的指标低于FKNN-DPC算法,但在其余两个UCI数据集上的聚类指标均优于全部对比算法,尤其在属性数量较多的数据集Soybean(Small)上和Statlog(Heart)上,算法根据近邻样本所属类簇的众数分配样本的策略有效利用了多个属性提供的维度信息来判断近邻样本间的相似性,使得DCM-DPC算法的聚类指标相对对比算法更具有明显的优势。

表4展示了6种算法在人工数据集上的聚类结果,其中加粗字体表示较优的实验结果。实验结果显示,DCM-DPC算法在参与测试的各个数据集上的聚类指标都较优秀,且比较平稳。在Aggregation、Jain、Spiral和R15人工数据集上,DCM-DPC算法的聚类指标优于或持平对比算法,并在Jain和Spiral数据集上实现了零差错;在D31和Flame数据集中指标分别略低于FKNN-DPC算法、SNN-DPC算法和DPC算法。

表3 6种算法在UCI数据集上的聚类性能

Table 3 Clustering performance of 6 algorithms on UCI datasets

Algorithm	Iris			Soybean(Small)			Statlog(Heart)		
	AMI	ARI	FMI	AMI	ARI	FMI	AMI	ARI	FMI
DCM-DPC	0.786 9	0.815 7	0.874 2	**0.957 7**	**0.959 8**	**0.970 0**	**0.263 2**	**0.351 2**	**0.673 1**
DPC	0.724 7	0.703 7	0.803 2	0.662 6	0.559 1	0.666 9	0.027 9	0.019 8	0.533 4
FKNN-DPC	**0.883 1**	**0.903 8**	**0.935 5**	0.662 6	0.559 1	0.666 9	0.073 6	0.107 0	0.579 1
SNN-DPC	0.540 5	0.482 6	0.695 7	0.555 8	0.474 1	0.672 3	0.241 7	0.266 0	0.639 7
DBSCAN	0.640 1	0.612 0	0.729 1	0.807 9	0.661 0	0.781 3	0.035 7	0.066 0	0.571 1
K-means++	0.711 1	0.692 6	0.800 6	0.873 3	0.820 1	0.864 4	0.015 7	0.027 6	0.527 0

表4 6种算法在人工数据集上的聚类性能

Table 4 Clustering performance of 6 algorithms on artificial datasets

Algorithm	Aggregation			D31			Flame		
	AMI	ARI	FMI	AMI	ARI	FMI	AMI	ARI	FMI
DCM-DPC	**0.995 6**	**0.997 8**	**0.998 3**	0.960 8	0.946 1	0.947 8	0.970 4	0.988 1	0.994 5
DPC	0.992 2	0.995 6	0.996 6	0.955 4	0.936 5	0.938 5	**1.000 0**	**1.000 0**	**1.000 0**
FKNN-DPC	0.990 5	0.994 9	0.996 0	**0.965 4**	**0.952 3**	**0.953 8**	0.926 7	0.966 7	0.984 5
SNN-DPC	0.950 0	0.959 4	0.968 1	0.964 2	0.950 9	0.952 5	0.897 4	0.950 1	0.976 8
DBSCAN	0.968 1	0.977 9	0.982 7	0.903 2	0.809 5	0.816 3	0.866 5	0.938 8	0.971 2
K-means++	0.807 6	0.741 1	0.792 0	0.926 6	0.852 5	0.858 3	0.430 4	0.490 7	0.755 1
Algorithm	Jain			Spiral			R15		
	AMI	ARI	FMI	AMI	ARI	FMI	AMI	ARI	FMI
DCM-DPC	**1.000 0**	**1.000 0**	**1.000 0**	**1.000 0**	**1.000 0**	**1.000 0**	**0.993 8**	**0.992 8**	**0.993 3**
DPC	0.618 3	0.714 6	0.881 9	**1.000 0**	**1.000 0**	**1.000 0**	**0.993 8**	**0.992 8**	**0.993 3**
FKNN-DPC	0.709 2	0.822 4	0.935 9	**1.000 0**	**1.000 0**	**1.000 0**	**0.993 8**	**0.992 8**	**0.993 3**
SNN-DPC	0.466 7	0.514 6	0.790 5	**1.000 0**	**1.000 0**	**1.000 0**	**0.993 8**	**0.992 8**	**0.993 3**
DBSCAN	0.928 1	0.975 8	0.990 6	**1.000 0**	**1.000 0**	**1.000 0**	0.983 2	0.975 8	0.979 9
K-means++	0.491 6	0.584 7	0.820 5	−0.005 5	−0.006 0	0.327 4	0.943 8	0.897 1	0.905 1

图5~图10展示了6种算法在人工数据集上的聚类效果，其中*K*-means++算法选取实验聚类指标最优结果。不同类簇的样本以及离散点分别用不同的颜色表示。在同一组聚类效果对比图中，代表不同聚类算法的图片之间相同的颜色表示对应于同一个类簇。

图5显示了6种算法对Aggregation数据集的聚类结果。除了*K*-means++算法，其余5种算法都在Aggregation数据集上取得了较好的聚类效果。但DBSCAN算法将左上角类簇右边缘部分样本和右侧两个类簇的邻接处样本误判成了离散点。在数据集左右两侧类簇的2处边缘样本纠缠处，SNN-DPC算法错误分配了17个样本，DPC和FKNN-DPC算法分别错误分配了2个样本；DCM-DPC算法仅有1个样本分配错误，聚类效果最好，对边缘样本的分配也是最准确的。

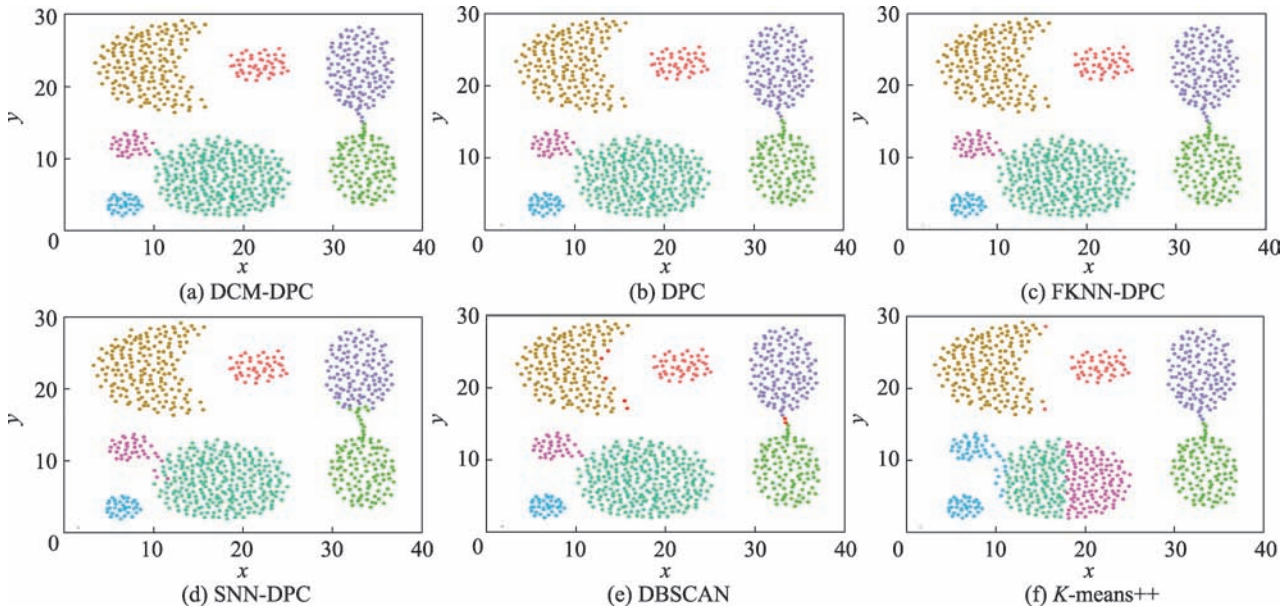

图5 Aggregation数据集聚类效果

Fig.5 Clustering effect on Aggregation dataset

图6显示了6种算法对D31数据集的聚类结果。D31数据集特点是规模较大,大部分样本聚合比较紧密,多处边缘样本存在纠缠,也有少数较为离散的样本。6种算法聚类指标相差不大,DBSCAN算法将大量类簇边缘样本误判成了离散点,指标最低;DCM-DPC算法对相互纠缠的边缘样本判断较准确,但在聚类过程中将右侧距离类簇较远的3个样本误判成了离散点,使聚类指标略低于FKNN-DPC和SNN-DPC算法。此外,D31数据集中部分类簇存在少量样本深入到其他类簇的样本中,被其他类簇的样本包围,6种聚类算法在此都进行了不同程度的误判,导致聚类效果有所下降。

图7的Flame数据集特点是一个类簇半包围着另一个类簇。除了*K*-means++算法因其球形聚类特征使得聚类效果最差外,其余5种算法在数据集上都取得了良好的聚类效果,其中DPC算法聚类效果最优。DCM-DPC算法在两个类簇交界处的样本划分非常准确,而FKNN-DPC、SNN-DPC和DBSCAN算法则在分配边界样本时都出现了错误。但DCM-DPC算法将左上侧2个远离类簇的样本误判成了离散值,导致聚类指标略低于DPC算法。

图8的数据集Jain是两个月牙状的类簇相互咬合。DPC、FKNN-DPC、SNN-DPC和*K*-means++算法在类簇咬合处都出现了大量样本分配错误,因此聚类指标较差。DBSCAN算法则在聚类中心个数的确定上出现失误,将数据划分成了3类。DCM-DPC算法对咬合处样本的分配依旧非常准确,并实现了聚类结果零差错。

图9展示了6种算法对Spiral数据集的聚类结果。该数据集由三组相距明显的漩涡状类簇组成,类簇内部样本相邻紧密,类簇间样本相距较远,边界清晰,非常适合于密度聚类。除*K*-means++算法外,其余5种算法都准确无误地完成了聚类。

图10展示了6种算法对R15数据集的聚类结果。该数据集由15个类簇组成,外圈类簇间隔明显,内圈类簇则相互纠缠。DCM-DPC、DPC、FKNN-DPC和SNN-DPC算法聚类效果优于DBSCAN和*K*-means++算法,且对内圈类簇边缘纠缠的样本归属判断准确度都较高。由于内圈的类簇中存在样本深入到其他类簇中,被其他类簇的样本包围,导致6种算法均在此出现了误判。

实验结果表明,DCM-DPC算法在UCI数据集Soybean(Small)和Statlog(Heart)的各项聚类指标均优于对比算法,在Iris的聚类指标仅低于FKNN-DPC算法,且在属性较多的Soybean(Small)和Statlog(Heart)数据集上得益于多属性带来的丰富信息,聚类指标更加突出。在人工数据集Aggregation、Jain、Spiral和R15上,DCM-DPC算法的三个指标均优于

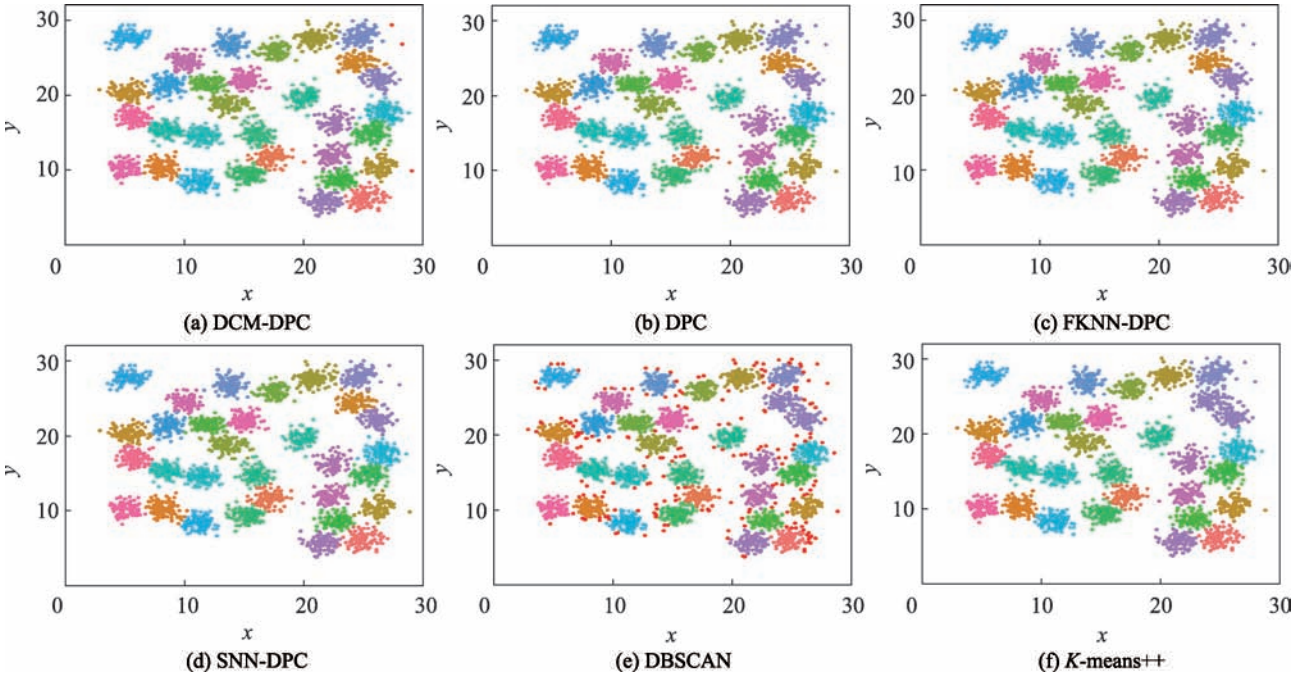

图6 D31数据集聚类效果

Fig.6 Clustering effect on D31 dataset

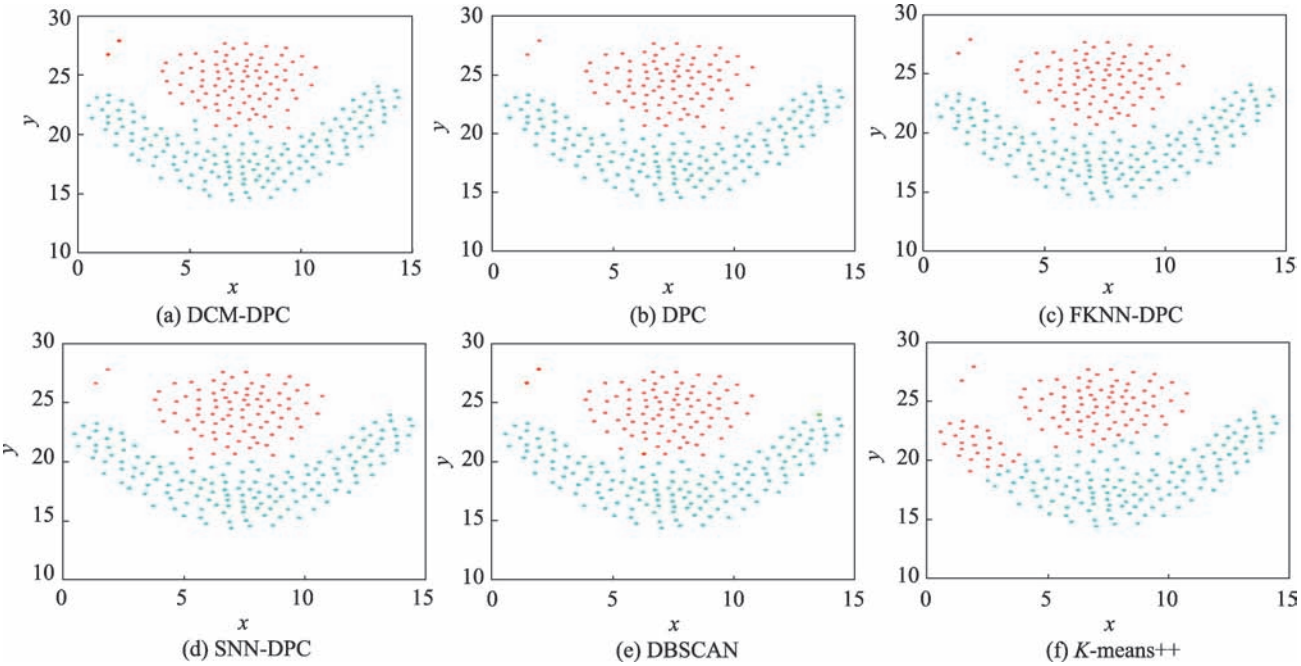

图7 Flame数据集聚类效果

Fig.7 Clustering effect on Flame dataset

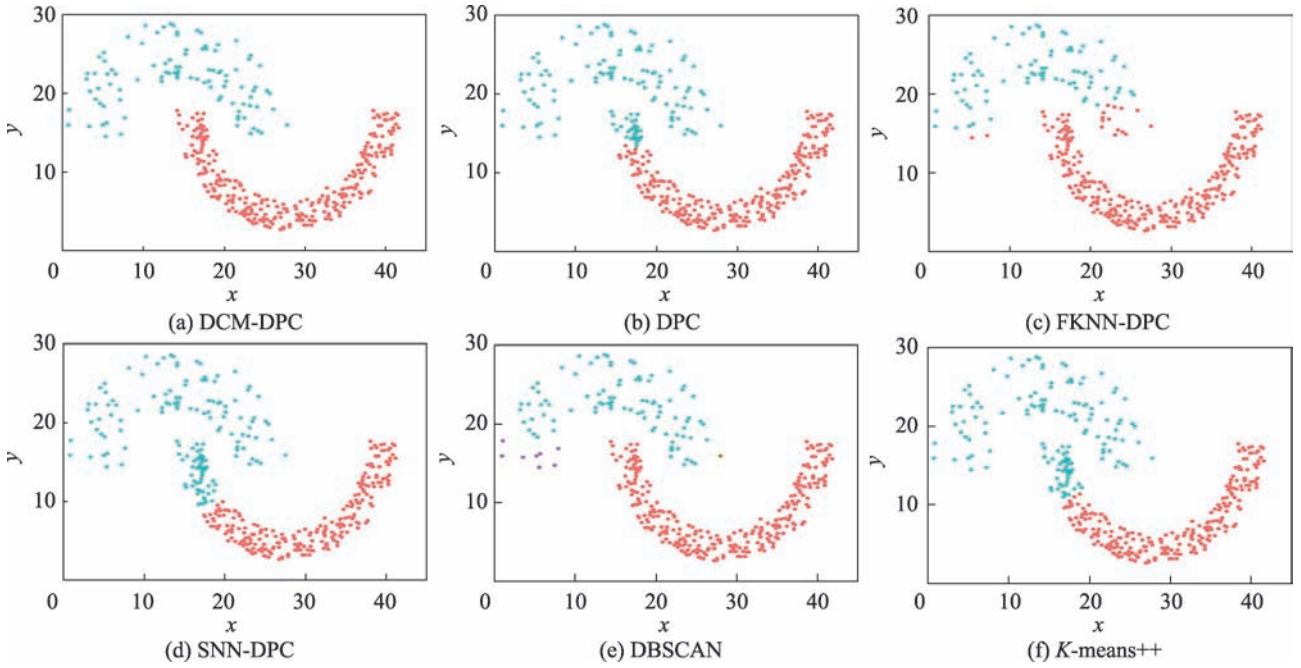

图8 Jain数据集聚类效果

Fig.8 Clustering effect on Jain dataset

或等于对比算法。但由于DCM-DPC算法在离散样本的判定上较为严格，可能会造成误判，这也是算法在数据集D31上指标略低于FKNN-DPC和SNN-DPC算法，在数据集Flame上指标略低于DPC算法的主要原因。

综合来看，DCM-DPC算法在不同规模和属性数的数据集上都有良好的表现，对数据的适应广泛，并具有良好的鲁棒性。特别是对边界相互纠缠或咬合的类簇，能精确地分配其边界样本，相对于对比算法具有明显优势。

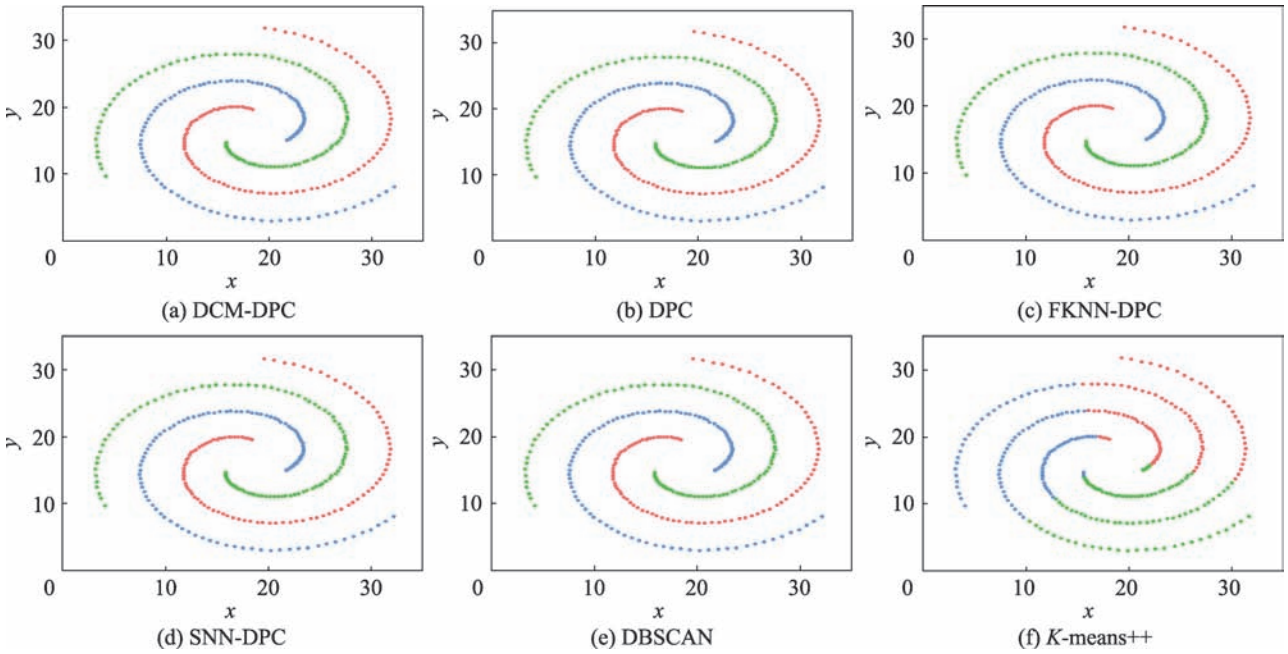

图9 Spiral数据集聚类效果

Fig.9 Clustering effect on Spiral

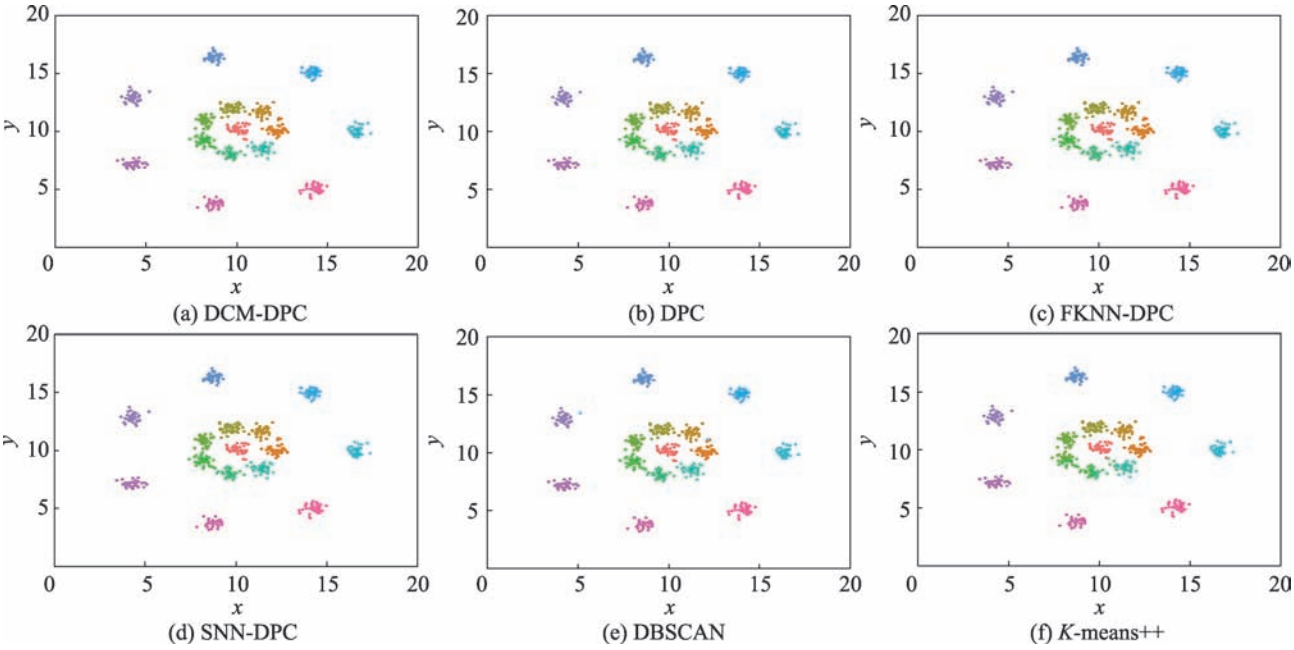

图10 R15数据集聚类效果

Fig.10 Clustering effect on R15 dataset

4 结束语

本文尝试提出了一种去中心化加权簇归并的密度峰值聚类算法DCM-DPC。DPC算法依托聚类中心点聚类的方法容易影响聚类效果,且聚类中心点的选择需要人为干预。对此本文提出了消除聚类中心点的核心样本组聚类方法,通过由位于较高局部密度且互为近邻的样本组成的核心样本组形成类簇雏形,并取代聚类中心点成为其余样本划分的依据。核心样本组较聚类中心更加稳定,能使聚类具有更好的鲁棒性。新定义的局部密度更好地描述了数据的内部结构,使本文算法可以在不同规模、属性数和类簇的数据集上得到良好的聚类结果;通过样

本的近邻点所属类簇的众数来决定样本归属，使样本划分时与类簇的关联性更强，有效缓解了跟随错误的产生。在人工和UCI数据集上的实验显示，本文算法在同类算法中具有较好的表现，且较对比算法能更加精确地分配相互纠缠或咬合的类簇的边界样本。由于本文算法在离散值的判定上比较严格，可能对游离的样本产生误判，提高对离散点的识别将是下一步的研究方向。

参考文献：

[1] ROSENBERGER C, CHEHDI K. Unsupervised clustering method with optimal estimation of the number of clusters: application to image segmentation[C]//Proceedings of the 15th International Conference on Pattern Recognition, Barcelona, Sep 3-8, 2000. Washington: IEEE Computer Society, 2000: 1656-1659.

[2] WANG C D, LAI J H, HUANG D, et al. SVStream: a support vector-based algorithm for clustering data streams[J]. IEEE Transactions on Knowledge and Data Engineering, 2013, 25 (6): 1410-1424.

[3] 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 Y W, SHEN L L, ZHONG C M, et al. Survey on density peak clustering algorithm[J]. Journal of Computer Research and Development, 2020, 57(2): 378-394.

[4] DING J J, HE X X, YUAN J Q, et al. Automatic clustering based on density peak detection using generalized extreme value distribution[J]. Soft Computing, 2018, 22(9): 2777-2796.

[5] ABE K, MINOURA K, MAEDA Y, et al. Model-based clustering for flow and mass cytometry data with clinical information [J]. BMC Bioinformatics, 2020, 21(13): 393.

[6] XIA S Y, PENG D W, MENG D Y, et al. A fast adaptive K-means with no bounds[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020: 1-13.

[7] POTHULA K R, SMYRNOVA D, SCHRODER G F. Clustering cryo-EM images of helical protein polymers for helical reconstructions[J]. Ultramicroscopy, 2019, 203: 132-138.

[8] BARANWAL M, SALAPAKA S. Clustering and supervisory voltage control in power systems[J]. International Journal of Electrical Power & Energy Systems, 2019, 109: 641-651.

[9] 陆川伟, 孙群, 季晓林, 等. 一种基于核距离的车辆轨迹点聚类方法[J]. 武汉大学学报(信息科学版), 2020, 45(7): 1082-1088.
LU C W, SUN Q, JI X L, et al. A method of vehicle trajectory points clustering based on kernel distance[J]. Geomatics and Information Science of Wuhan University, 2020, 45(7): 1082-1088.

[10] YAN X Q, YE Y D, QIU X Y, et al. Synergetic information bottleneck for joint multiview and ensemble clustering[J]. Information Fusion, 2020, 56: 15-27.

[11] 柏锷湘, 罗可, 罗潇. 结合自然和共享最近邻的密度峰值聚类算法[J]. 计算机科学与探索, 2021, 15(5): 931-940.
BAI E X, LUO K, LUO X. Peak density clustering algorithm combining natural and shared nearest neighbor[J]. Journal of Frontiers of Computer Science and Technology, 2021, 15 (5): 931-940.

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

[13] 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.

[14] SEYED A S, ABDULRAHMAN L, PARHAM M, et al. Dynamic graph-based label propagation for density peaks clustering[J]. Expert Systems with Applications, 2019, 115: 314-328.

[15] 丁世飞, 徐晓, 王艳茹. 基于不相似性度量优化的密度峰值聚类算法[J]. 软件学报, 2020, 31(11): 3321-3333.
DING S F, XU X, WANG Y R. Optimized density peaks clustering algorithm based on dissimilarity measure[J]. Journal of Software, 2020, 31(11): 3321-3333.

[16] LIU R, WANG H, YU X M. Shared-nearest-neighbor-based clustering by fast search and find of density peaks[J]. Information Sciences, 2018, 450: 200-226.

[17] 王大刚, 丁世飞, 钟锦. 基于二阶 k 近邻的密度峰值聚类算法研究[J]. 计算机科学与探索, 2021, 15(8): 1490-1500.
WANG D G, DING S F, ZHONG J. Research of density peaks clustering algorithm based on second-order k neighbors [J]. Journal of Frontiers of Computer Science and Technology, 2021, 15(8): 1490-1500.

[18] 王芙银, 张德生, 张晓. 结合鲸鱼优化算法的自适应密度峰值聚类算法[J]. 计算机工程与应用, 2021, 57(3): 94-102.
WANG F Y, ZHANG D S, ZHANG X. Adaptive density peaks clustering algorithm combining with whale optimization algorithm[J]. Computer Engineering and Applications, 2021, 57(3): 94-102.

[19] 纪霞, 姚晟, 赵鹏. 相对邻域与剪枝策略优化的密度峰值聚类算法[J]. 自动化学报, 2020, 46(3): 562-575.
JI X, YAO S, ZHAO P. Relative neighborhood and pruning strategy optimized density peaks clustering algorithm[J]. Acta Automatica Sinica, 2020, 46(3): 562-575.

[20] 陆佳炜, 吴涵, 张元鸣, 等. 融合功能语义关联计算与密度峰值检测的Mashup服务聚类方法[J]. 计算机学报, 2021, 44(7): 1501-1516.
LU J W, WU H, ZHANG Y M, et al. Mashup service

clustering method via integrating functional semantic association calculation and density peak detection[J]. Chinese Journal of Computers, 2021, 44(7): 1501-1516.

[21] 刘娟, 万静. 自然反向最近邻优化的密度峰值聚类算法[J]. 计算机科学与探索, 2021, 15(10): 1888-1899.
LIU J, WAN J. Optimized density peak clustering algorithm by natural reverse nearest neighbor[J]. Journal of Frontiers of Computer Science and Technology, 2021, 15(10): 1888-1899.

[22] LIU Y H, MA Z M, YU F. Adaptive density peak clustering based on K-nearest neighbors with aggregating strategy[J]. Knowledge-Based Systems, 2017, 133: 208-220.

[23] SUN L P, TAO T, ZHENG X Y, et al. Combining density peaks clustering and gravitational search method to enhance data clustering[J]. Engineering Applications of Artifical Intelligence, 2019, 85: 865-873.

[24] HUANG D, WANG C D, WU J S, et al. Ultra-scalable spectral clustering and ensemble clustering[J]. IEEE Transactions on Knowledge and Data Engineering, 2020, 32(6): 1212-1226.

[25] WANG S L, LI Q, ZHAO C F, et al. Extreme clustering—a clustering method via density extreme points[J]. Information Sciences, 2021, 542: 24-39.

[26] 朱庆峰, 葛洪伟. *K* 近邻相似度优化的密度峰聚类[J]. 计算机工程与应用, 2019, 55(2): 148-153.
ZHU Q F, GE H W. Density peaks clustering optimized by *K* nearest neighbor's similarity[J]. Computer Engineering and Applications, 2019, 55(2): 148-153.

[27] 陈磊, 吴润秀, 李沛武, 等. 加权K近邻和多簇合并的密度峰值聚类算法[J/OL]. 计算机科学与探索(2021-04-19) [2021-12-16]. https://kns.cnki.net/kcms/detail/11.5602.tp.20210416.1428.002.html.
CHEN L, WU R X, LI P W, et al. Weighted K-nearest neighbors and multi-clusters merge density peaks clustering algorithm[J/OL]. Journal of Frontiers of Computer Science and Technology (2021-04-19) [2021-12-16]. https://kns.cnki.net/kcms/detail/11.5602.tp.20210416.1428.002.html.

[28] 杜浩翠, 谢维信. 基于改进的密度峰值聚类的扩展目标跟踪算法[J]. 信号处理, 2021, 37(5): 735-746.
DU H C, XIE W X. Extended target tracking algorithm based on improved density peak clustering[J]. Journal of Signal Processing, 2021, 37(5): 735-746.

[29] 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.

[30] 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, Portland, 1996. Menlo Park: AAAI, 1996: 226-231.

[31] ARTHUR D, VASSILVITSKII S. K-means++: the advantages of careful seeding[C]//Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, New Orleans, Jan 7-9, 2007. New York: ACM, 2007: 1027-1035.

[32] NGUYEN X V, 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.

[33] 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.

赵力衡(1976—),男,四川成都人,硕士,高级工程师,CCF专业会员,主要研究方向为数据挖掘、海量数据存储。

ZHAO Liheng, born in 1976, M.S., senior engineer, professional member of CCF. His research interests include data mining and massive data storage.

王建(1979—),男,四川泸州人,博士,副教授,主要研究方向为人工智能、数据挖掘。

WANG Jian, born in 1979, Ph.D., associate professor. His research interests include artificial intelligence and data mining.

陈虹君(1979—),女,四川广安人,硕士,教授,主要研究方向为大数据、人工智能。

CHEN Hongjun, born in 1979, M.S., professor. Her research interests include big data and artificial intelligence.