文章编号:1671-9352(2024)03-0027-10 DOI:10.6040/j.issn.1671-9352.4.2023.040

模糊边界剥离聚类

孙嘉睿,杜明晶
(江苏师范大学计算机科学与技术学院, 江苏 徐州 221100)

摘要:提出了一种模糊边界剥离聚类(fuzzy border-peeling clustering, FBP)算法。首先,采用了一种基于 Cauchy 核的动态密度估计方式来计算数据点密度;然后,使用逐层剥离策略区分边界数据和核心数据;接着,利用核心数据间的可达性实现核心区域聚类;最后,采用模糊分配策略实现边界数据的软划分。在人工数据集和真实数据集上与 10 种算法(包含 6 种密度聚类算法和 4 种模糊聚类算法)作了对比。实验结果表明,在所有数据集上,FBP 的调整兰德系数 ARI 指标平均提高了 21%~60%,FBP 的标准化互信息 NMI 指标平均提升了 12%~47%,基于 Cauchy 核和模糊分配策略优化后的边界剥离聚类算法显著提高了聚类的准确性。

关键词:密度聚类;边界剥离聚类;模糊聚类;软化分;柯西核函数
中图分类号:TP18 **文献标志码**:A
引用格式:孙嘉睿,杜明晶. 模糊边界剥离聚类[J]. 山东大学学报(理学版),2024,59(3):27-36,50.

Fuzzy border-peeling clustering

SUN Jiarui, DU Mingjing
(School of Computer Science and Technology, Jiangsu Normal University, Xuzhou 221100, Jiangsu, China)

Abstract: A fuzzy border-peeling clustering (FBP) algorithm is proposed. First, a density estimation method based on Cauchy kernel is used to calculate the densities of data points. Secondly, the boundary data are separated from the core data using the layer-by-layer peeling strategy. Thirdly, the reachability between the core data is used to achieve the core region clustering. Finally, a fuzzy assignment strategy is used to achieve the soft partitioning of the boundary data. A comparison is made between the fuzzy border-peeling clustering and 10 benchmark algorithms, including 6 density-based clustering algorithms and 4 fuzzy clustering algorithms, on artificial and real-world datasets. The experimental results show that on all datasets, FBP has the ARI (adjusted rand index) increased by 21% to 60% on average, and FBP has the NMI (normalized mutual information) increased by 12% to 47% on average. The border-peeling clustering algorithm optimized based on Cauchy kernel and fuzzy assignment strategy significantly improves the accuracy of clustering.

Key words: density-based clustering; border-peeling clustering; fuzzy clustering; soft clustering; Cauchy kernel function

0 引言

聚类可以在无标注的情况下实现数据的有效划分,因此,该技术被广泛应用于模式识别和数据挖掘中。聚类算法可以大致分为基于划分的聚类、层次聚类、基于模型的聚类和基于密度的聚类。相较于其他聚类范式,基于密度的聚类方法是在更高的粒度级中探索数据的分布情况,将簇假设为具有较高密度的连通区域,而将稀疏区域中的数据视为噪声[1]。在该假设的基础上,基于密度的聚类能够在不给定簇数目的情况下有效识别任意形状的簇,同时具有良好的抗噪能力。基于密度的聚类算法已在生物学、地质学等领域得到了广

收稿日期:2023-06-02; **网络出版时间**:2023-11-28 08:59:06
网络出版地址:https://link.cnki.net/urlid/37.1389.n.20231124.2154.004
基金项目:国家自然科学基金资助项目(62006104);江苏师范大学研究生科研创新项目(2022XKT1528)
第一作者:孙嘉睿(1997—),男,硕士研究生,研究方向为机器学习和数据挖掘. E-mail:sunjr@jsnu.edu.cn
*通信作者:杜明晶(1989—),男, 副教授,博士,研究方向为数据挖掘和三支决策. E-mail:dumj@jsnu.edu.cn

泛应用[2]。目前,经典的密度聚类算法包括 DBSCAN(density-based spatial clustering of applications with noise)[3]、HDBSCAN(hierarchical density-based spatial clustering of applications with noise)[4]、密度峰值聚类(density peaks clustering, DPC)[5]等。

DBSCAN[3]是最著名的密度聚类算法,能够在类数目未知的情况下发现任意形状的簇,因此得到了学术界和工业界的广泛关注[6-7]。HDBSCAN[4]继承了 DBSCAN 识别任意形状簇的能力,通过引入层级结构从而具备了处理可变密度数据的能力。相较于 DBSCAN 算法,DPC[5]具有更少的参数设置,该算法首先根据数据的局部密度和相对距离快速确定类簇中心,然后将剩余数据分配到密度更高且距其最近的数据所在的簇。由于其简单且易实现,研究者们对密度峰值聚类算法展开了大量的研究[8-10]。

Averbuch-Elor 等[11]提出一种新的密度聚类算法,名为边界剥离聚类(border-peeling clustering, BP)。边界剥离聚类的核心思想是通过迭代剥离簇边界区域上的数据点,直到发现各簇的核心区域,之后利用类 DBSCAN 算法完成核心数据点的划分,并根据剥离阶段所建立的连接完成边界数据的分配。因为在真实应用场景中数据分布结构复杂,簇与簇的边界往往难以区分,而边界剥离聚类和其他密度聚类算法类似,仅能给出一种硬划分的聚类结果,所以难以柔性表示数据的归属情况。

针对上述问题,本文提出了一种模糊边界剥离聚类(fuzzy border-peeling clustering, FBP)算法。本算法首先使用一种 Cauchy 核密度估计度量数据密度,然后使用逐层剥离策略区分边界数据和核心数据,利用核心数据间的可达性实现核心区域聚类,最后利用边界数据的近邻信息实现软划分。在人工数据集和真实数据集上的实验结果表明,本算法优于多个基准算法,其在边界模糊、分布复杂的数据上具有不错表现。

1 相关工作

1.1 基于密度的聚类

Ankerst 等[12]提出了 OPTICS(ordering points to identify the clustering structure)以改进 DBSCAN 算法,OPTICS 引入核心距离和可达距离的概念构建数据的层次结构,并通过一个全局阈值实现类簇的划分。HDBSCAN[4]算法作为 DBSCAN 在层次结构上的另一种扩展,借鉴了 OPTICS 中的可达概念,提出了基于互可达数据关联关系,将互可达图演化为最小生成树结构,通过该结构构建簇的层级关系。OPTICS 和 HDBSCAN 在可变密度数据上都有着不错的表现,但是这 2 种聚类算法难以识别具有重叠区域的簇。

DPC[5]算法的核心思想是通过密度与距离的关系找出核心点,并将非核心点分配给距离其最近的稠密点,这使得 DPC 算法适用于具有高斯分布特性的数据(簇中心稠密、簇边界稀疏)。由于其采用的是一种链式分配方式,所以一旦有一个数据点分配错误,则后续数据点也会出现同样的划分错误,这种错误传播会导致聚类质量下降。为了避免上述情况所带来的负面效应,DPC-CE[13]基于图来评估局部中心之间的连接信息,然后将这些信息整合到距离计算和数据分配中,从而避免错误传播的发生。尽管上述策略可以有效防止发生错误传播,但这些算法仍然无法较好地聚类存在多密度中心的簇。VDPC[14]使用初步筛选代表点的方法构建初始聚类,从而系统地分析数据的分布情况,以获得更加准确的聚类结果。

BP[11]是一种新型聚类范式,能通过剥离策略将数据划分为核心数据和边界数据,并采用不同的方法分别完成这 2 类数据的聚类,但该算法依然存在过度划分的问题。针对边界剥离聚类算法过度划分的问题,Du 等[15]提出了一种鲁棒型的边界剥离聚类(ROBP)算法,针对原始 BP 算法过度划分的问题,引入共享近邻分配策略将错误拆分的子簇重新合并,进而提高聚类精度。与 ROBP 的改进方向完全不同,FBP 算法为解决 BP 算法中硬划分策略所带来的问题,设计了一种面向剥离层级结构的模糊划分方式,以此提升算法的聚类精准度。陈延伟等[16]为提高 BP 算法处理可变密度数据的能力,提出了一种基于边界点检测的变密度聚类(VDCBD)算法,该算法将相对密度度量引入到原始算法中,可以有效识别密度不均匀的簇。在处理可变密度数据度量上与 VDCBD 略有不同,FBP 算法引入 Cauchy 核形式的密度估计方法用于密度度量。张柏恺等[17]提出了一种基于自然邻居思想的边界剥离聚类(NaN-BP)算法,该算法针对邻域参数自适应的问题,将自然邻居思想引入到边界剥离聚类中,消除了原始算法中的近邻参数敏感性问题。FBP 算法聚焦于算法的精度问题,而 NaN-BP 算法聚焦于参数敏感性问题。当前已有的改进算法并未尝试将边界剥离聚类算法扩展到软划分聚类上。

1.2　模糊聚类

FCM(fuzzy c-means)[18]是模糊聚类算法中运用最广的算法之一,它通过引入隶属度的概念,允许数据隶属于多个簇。研究者们开发了诸多该类算法的变种,包括 MEC(maximum entropy clustering)[19]、FSC(fuzzy subspace clustering)[20]和 FCMFP(FCM with focal point)[21]等。Krishnapuram 等[22]提出了 PCM(possibilistic c-means algorithm),相较于 FCM,该算法在具有噪声数据的情况下表现更好,但同样对中心点的初始化选择较为敏感。为了识别非球形簇,KFCM(kernel fuzzy c-means)[23]在 FCM 算法中引入了核函数的概念,通过核函数将数据从原始空间映射到更高维空间中;MKFC(multiple kernel fuzzy c-means)[24]运用多核学习的方式扩展 FCM 算法。尽管这 2 种算法具备了处理非球形簇的能力,但当数据分布非常复杂时,KFCM 和 MKFC 也难以取得较好效果。此外,上述模糊聚类是基于 k-means 范式的,而非本文采用的密度聚类范式。

2　边界剥离聚类

2.1　概念与符号

本小节给出边界剥离聚类和模糊边界剥离聚类相关符号。假设 $X=\{x_1,x_2,\cdots,x_n\}$ 为由 n 个数据构成的数据集。$d(x_i,x_j)$ 表示任意 2 个数据点 x_i 和 x_j 间的距离。

令 $X^{(t)}$ 为第 t 次边界剥离所余数据集合,有 $X^{(t)}\subset X$。

$NN_k^{(t)}(x_i)$ 表示数据点 $x_i\in X^{(t)}$ 在集合 $X^{(t)}$ 中第 k 个最近邻数据,因此有 $NN_k^{(t)}(x_i)\in X^{(t)}$。

$N_k^{(t)}(x_i)$ 表示数据点 $x_i\in X^{(t)}$ 在集合 $X^{(t)}$ 中前 k 个近邻组成的集合,因此有 $N_k^{(t)}(x_i)\subset X^{(t)}$。

$RN_k^{(t)}(x_i)$ 表示数据点 $x_i\in X^{(t)}$ 在集合 $X^{(t)}$ 中反 k 近邻组成的集合,其形式如下:

$$RN_k^{(t)}(x_i)=\{x_j\mid x_i\in N_k^{(t)}(x_j)\},\tag{1}$$

其中 $RN_k^{(t)}(x_i)\subset X^{(t)}$,即 x_i 是所有 x_j 的 k 近邻。

表 1 总结了上述符号的概念。

表 1　符号摘要

Table 1　Summary of notations

符号	概念
n	数据点数目
k	近邻数目
t	第 t 次剥离
x_i	任意数据点
X	数据集合
$X^{(t)}$	第 t 次剥离,剩余数据集合
$NN_k^{(t)}(x_i)$	第 t 次剥离,数据点 x_i 的第 k 近邻
$N_k^{(t)}(x_i)$	第 t 次剥离,数据点 x_i 的 k 近邻集合
$RN_k^{(t)}(x_i)$	第 t 次剥离,数据点 x_i 的反 k 近邻集合

2.2　边界剥离的基本原理

边界剥离聚类给出了一个重要的假设:簇的边界区域包裹着簇的核心区域,且各个簇的核心区域被其低密度的边界区域所分隔。边界剥离聚类认为簇边界区域的密度往往低于其核心区域的密度,因此可以通过逐层剥离策略检测边界区域和核心区域,进而实现相邻簇的划分。边界剥离聚类算法大致分为 3 步:(1)边界数据检测,通过逐层边界剥离策略检测确定边界数据和核心数据;(2)边界数据分配,在边界剥离过程中,建立边界数据与核心数据的隶属关系;(3)核心数据聚类,在边界剥离结束后,利用类 DBSCAN 算法实现核心数据聚类。

3　模糊边界剥离聚类

以下介绍 FBP 算法的主要模块及算法步骤。

3.1　边界数据检测

3.1.1　密度估计

核密度估计(kernel density estimation, KDE)是一种非参数统计方法,用于估计概率密度函数(probability density function, PDF)的形状和分布,是一种被广泛使用的密度估计方法。点 x_i 的核密度估计器如下:

$$\rho_i=\sum_{x_j\in V}\kappa\left(\frac{\|x_j-x_i\|}{h}\right),\tag{2}$$

其中:$\kappa(\cdot)$是一个非负的单调递减函数;h 是用于控制影响范围的带宽项;变量 x_i 和 x_j 分别代表一个测试实

例和一个样本。

Cauchy 密度核(Cauchy density kernel)是一种以 Cauchy 分布为基础、在核密度估计中使用的核函数。给定一个数据点 x,Cauchy 密度核的定义为

$$K(x;x_0,\gamma)=1\Big/\left(\pi*\gamma*\left(\frac{\|x-x_0\|^2}{\gamma^2}+1\right)\right), \tag{3}$$

其中:x_0 是核函数的中心(通常是一个已知的参考点);γ 是核函数的尺度参数。需要注意的是,Cauchy 密度核相比于其他核函数(如高斯核)具有更宽的尾部,因此对于极端值和离群点的影响更大。

对于核函数的选择,原算法使用的是 Gaussian 密度核,FBP 在密度评估函数上对原始算法做了改进,将 Gaussian 核密度估计改为 Cauchy 核密度估计的形式。在引入 Cauchy 密度核的基础上,使用了基于 k 近邻的可变带宽,该带宽值会跟随数据点的局部稀疏程度自适应变化,使得算法可以更好地识别可变密度的簇,且具有良好的平滑性。数据点的初始密度估计定义为

$$b_i=\sum_{x_j\in RN_k(x_i)}1\Big/\left(\frac{\|x_i-x_j\|^2}{\|NN_k(x_j)-x_j\|^2}+1\right), \tag{4}$$

其中:求和项为 Cauchy 核函数形式;$NN_k(x_i)$ 为 x_i 在整个数据集中的第 k 个近邻点;$RN_k(x_i)$ 为数据点 x_i 在整个数据集中的反 k 近邻集合。

模糊边界剥离聚类与原始边界剥离聚类都采用逐层剥离策略,该策略会在每次剥离完成后重新计算所剩数据的密度情况。在第 t 次迭代中,数据点的密度计算公式如下:

$$b_i^{(t)}=\sum_{x_j\in RN_k^{(t)}(x_i)}1\Big/\left(\frac{\|x_i-x_j\|^2}{\|NN_k^{(t)}(x_j)-x_j\|^2}+1\right)。 \tag{5}$$

3.1.2 边界剥离

边界剥离的目标是通过动态密度评估方式逐层确定簇的边界数据点,并不断剥离边界数据,最终识别簇的核心数据点。

令 $B_i^{(t)}$ 为数据点 $x_i\in X^{(t)}$ 的剥离标识。在第 t 次迭代中,数据点 x_i 被判定为边界点,则其剥离标识设置为 1,即 $B_i^{(t)}=1$;否则,其剥离标识设置为 0,即 $B_i^{(t)}=0$。剥离标识的定义如下:

$$B_i^{(t)}=\begin{cases}1, & \text{if } b_i^{(t)}\leqslant\tau^{(t)},\\ 0, & \text{otherwise},\end{cases} \tag{6}$$

其中 $\tau^{(t)}$ 为第 t 次迭代的密度阈值。这一系列阈值由一种密度自适应机制自动给定[11]。

第 t 次迭代剥离的边界点集合为

$$X_B^{(t)}=\{x_i\in X^{(t)}\mid B_i^{(t)}=1\}。 \tag{7}$$

下次迭代,剩余数据点的集合为当次数据点集合与边界点集合的差集,形式如下:

$$X^{(t+1)}=X^{(t)}\setminus X_B^{(t)}。 \tag{8}$$

当连续 2 次迭代所剥离数据的平均密度值间差异较大时,剥离结束,此时剩余数据定义为核心数据。假设迭代在 T 次结束,则 $X^{(T+1)}$ 为核心数据集合,所有已剥离的边界数据点集合表示为

$$X_B=X_B^{(1)}\cup X_B^{(2)}\cup\cdots\cup X_B^{(T)}, \tag{9}$$

其中 $X_B^{(1)},X_B^{(2)},\cdots,X_B^{(T)}$ 为每次迭代剥离的边界点集合。

3.2 核心数据聚类

原始边界剥离聚类为获取各个核心数据连接阈值,在边界剥离阶段逐层获取所有边界数据与内层数据的连接距离,该方式需迭代式的更新连接阈值,其计算开销较大。FBP 算法在边界剥离阶段结束后,一次性计算核心数据,极大提升了计算效率。

核心数据点 $x_i\in X^{(T+1)}$ 的连接阈值定义为

$$l_i=\min\left\{\sum_{x_i\in N_k^B(x_i)}\frac{3}{k}d(x_i,x_j),\mathrm{mean}(D_k)+\mathrm{std}(D_k)\right\}, \tag{10}$$

其中:k 表示近邻数目;$N_k^B(x_i)$ 表示核心数据点 $x_i\in X^{(T+1)}$ 在边界数据点中的 k 近邻集合,即 $N_k^B(x_i)\subseteq X_B$;$D_k=\bigcup\limits_{x_i\in X}\{d(x_i,x_j)\mid x_j\in N_k(x_i)\}$;mean 表示该集合的平均值,std 表示该集合的标准差。

定义 1　对于任意的 2 个核心数据 x_i 和 x_j,若存在一个核心数据序列 $p_1,p_2,\cdots,p_S$(其中 S 为该序列的长度),满足 p_1 为 x_i,p_S 为 x_j,且满足对于任意的 $1\leqslant s\leqslant S$,有 $d(p_s,p_{s+1})\leqslant\max(l_i,l_j)$,即 p_s 和 p_{s+1}之间的距离小于等于两者连接阈值的最大值,则称由 x_i 到 x_j 可达。

通过互可达性质实现核心数据的划分,形成初始簇集合 $\hat{C}=\{\hat{C}_1,\hat{C}_2,\cdots,\hat{C}_K\}$。

3.3　边界数据分配

在模糊集合中,元素可以以不同的程度属于一个模糊集合,而不仅仅是二元的属于或不属于。为获得边界数据的模糊划分结果,FBP 算法反向遍历各已剥离的边界点集合:$X_B^{(T)},X_B^{(T-1)},\cdots,X_B^{(2)},X_B^{(1)}$。

令边界数据点$x_i\in X_B$到初始簇 $\hat{C}_c$ 的距离为

$$d(x_i,\hat{C}_c)=\min\{d(x_i,x_j)\mid x_j\in\hat{C}_c\}。\tag{11}$$

将边界数据点$x_i\in X_B$与簇 C_c 的隶属度关系定义为

$$m(x_i,C_c)=\frac{\dfrac{1}{d(x_i,\hat{C}_c)}}{\displaystyle\sum_{j=1}^{K}\frac{1}{d(x_i,\hat{C}_j)}}。\tag{12}$$

依据该隶属关系对边界数据点进行模糊分配,该隶属度关系表示了边界数据点 x_i 属于簇 C_c 的程度,取值范围通常是 0 到 1 之间,越接近 1 表示 x_i 越属于簇 C_c,反之则越不属于簇 C_c。

图 1 以一个例子形象地展示了 FBP 聚类算法的核心思想与处理流程。对于图 1 中 2 个靠近的簇,FBP 首先进行边界检测与剥离,根据式(5)估计每个数据点的密度,根据式(7)使用与 BP 类似的密度自适应机制来确定待剥离点,即将该层数据点密度正序排序后取第 10 个百分位数作为该层的剥离密度阈值 $\tau^{(t)}$,将密度小于 $\tau^{(t)}$ 的数据判断为边界点(标记为红色)并剥离。上述过程迭代进行,直到连续 2 次迭代所剥离数据的平均密度值间差异较大时,剥离结束,得到最终的核心点。然后根据式(10)计算各核心点的连接阈值,并根据定义 1 完成核心数据聚类,得到 2 个初始簇(标记为蓝色和黄色)。最后 FBP 算法根据式(12)由内到外逐层进行边界点的模糊划分(红色箭头由边界点指向其隶属度最高的簇),直到最外一层边界点完成划分,得到最终的聚类结果。

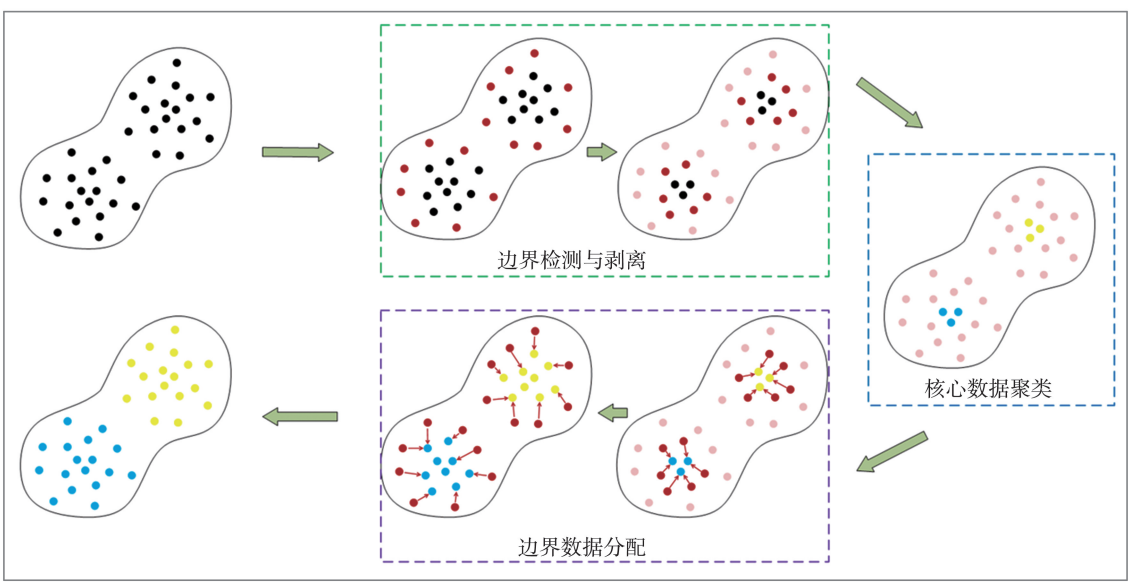

图 1　FBP 聚类算法流程

Fig.1　FBP clustering algorithm process

3.4　算法流程

模糊边界剥离聚类算法步骤如下。

算法 1　模糊边界剥离聚类

输入:数据集 X,近邻参数 k;

输出: 聚类结果 $C=\{C_1,C_2,\cdots,C_K\}$。

Step 1 $X^{(1)}\leftarrow X$;

Step 2 边界剥离;

Step 2.1 根据公式(5)计算 $X^{(t)}$ 中各数据点的密度值;

Step 2.2 根据式(6)判定 $X^{(t)}$ 中各数据点的剥离标识,然后根据式(7)确定被剥离的边界点集合;

Step 2.3 根据式(8)确定 $X^{(t)}$ 剥离后剩余的数据点集合 $X^{(t+1)}$;

Step 2.4 剥离结束至 Step 3,否则至 Step 2;

Step 3 根据公式(10)计算 $X^{(T+1)}$ 中各数据点的连接阈值;

Step 4 根据定义 1 完成 $X^{(T+1)}$ 的初始聚类;

Step 5 据公式(12)完成边界数据的模糊划分,得到最终的聚类结果。

4 实验分析

4.1 实验配置

为了验证本算法的有效性,将本算法与 10 种聚类算法进行比较,包括 BP[11]、DBSCAN[3]、HDBSCAN[4]、DPC[5]、DPC-CE[13]、VDPC[14]、FCM[18]、KFCM[23]、MKFC[24] 和 PCM[22]。BP 是 FBP 算法的原始算法,DBSCAN、HDBSCAN 和 DPC 都是经典的密度聚类算法,DPC-CE 和 VDPC 是 DPC 算法的改进算法,其余 4 种算法均为模糊聚类算法。本文对 FBP 和上述对比算法做如下参数配置,并从中选择最佳聚类结果。FBP 和 BP 算法的唯一参数是近邻数目 k,其参数取值范围为[5,50]。DBSCAN 有 2 个参数 ε 和 MinPts,ε 的取值范围为[wux:wuy5:5ux],MinPts 的取值范围为[5:1:50]。HDBSCAN 同样具有近邻数目 k,其取值范围如上。此外,HDBSCAN 还具有参数 m_c,其表示最小簇大小,取值范围为(5,10,15,20)。DPC、DPC-CE 和 VDPC 聚类都有截断阈值参数 d_c,其取值范围为[2%,10%]。VDPC 比前两者多了一个参数 δ,其取值范围为(1,2,3,4,5)。本文实验借鉴 DP-MD-FN[25] 算法中所使用的自动确定类簇中心的策略,通过引入调节参数 α 自动确定簇数目,取值范围为(0.5,1,1.5,2,2.5)。FCM、PCM、KFCM 和 MKFC 均具有参数 M,该参数的取值范围为(1.5,2.0,2.5,3.0,3.5,4.0,4.5,5.0)。KFCM比前三者多使用了一个参数 δ,其取值范围为(0.5,1.0,1.5,2.0,2.5,3.0)。由于 FCM、PCM、KFCM 和 MKFC 算法的表现均具有波动性,因此本实验在每个参数配置下重复执行了 100 次,取所有结果的中位数作为最终结果。FBP 和 BP 为确定性算法,无需重复执行。

表 2 实验数据集

Table 2 Experimental data sets

数据集名称	数据量	维数	类数
Jain	373	2	2
DD	1 300	2	3
Pathbased	300	2	3
Handl	715	2	3
SF	4 096	2	4
T4	7 236	2	6
Glass	214	10	6
Control	600	60	6
Dig	1 797	64	10
PageB	5 473	10	5
Optdigits	5 620	64	10
Crowd	10 845	28	6

本文选用了表 2 所示的 12 组数据集,其中包含 6 组人工数据集和 6 组真实数据集。6 组人工数据集分别为 Jain、DD、Pahtbased、Handl、SF、T4。6 组真实数据集分别是 Glass、Control、Dig、PageB、Optdigits、Crowd。

本文选用的数据集均具有类标签,因此,选取 2 种外部评价指标衡量算法的聚类表现。2 种指标分别是标准化互信息 NMI 和调整兰德系数 ARI,其中 NMI 的取值范围是[0,1],ARI 的取值范围是[-1,1]。两者均是数值越大表示聚类效果越好。

4.2 人工数据集结果

图 2 给出人工数据集上的实验结果,每一行为不同算法在同一数据集上生成的聚类结果,不同颜色的点代表着不同的簇,黑色点对应着噪声点。

Jain 数据集由 2 个具有不同密度的月牙形簇构成。从结果中不难发现:FBP 算法产生了正确的聚类结果;BP 算法产生了过度划分现象;DBSCAN 算法未识别出上方低密度的簇;HDBSCAN 给出了正确的簇数目,但将

稀疏边界数据点错识别为噪声数据;DPC 和 DPC-CE 准确识别了上方的簇,但将下方簇错误地分为诸多微簇;VDPC 算法同样产生了过度划分的现象;FCM、KFCM、MKFC 和 PCM 都将下方稠密的簇错分为多类。

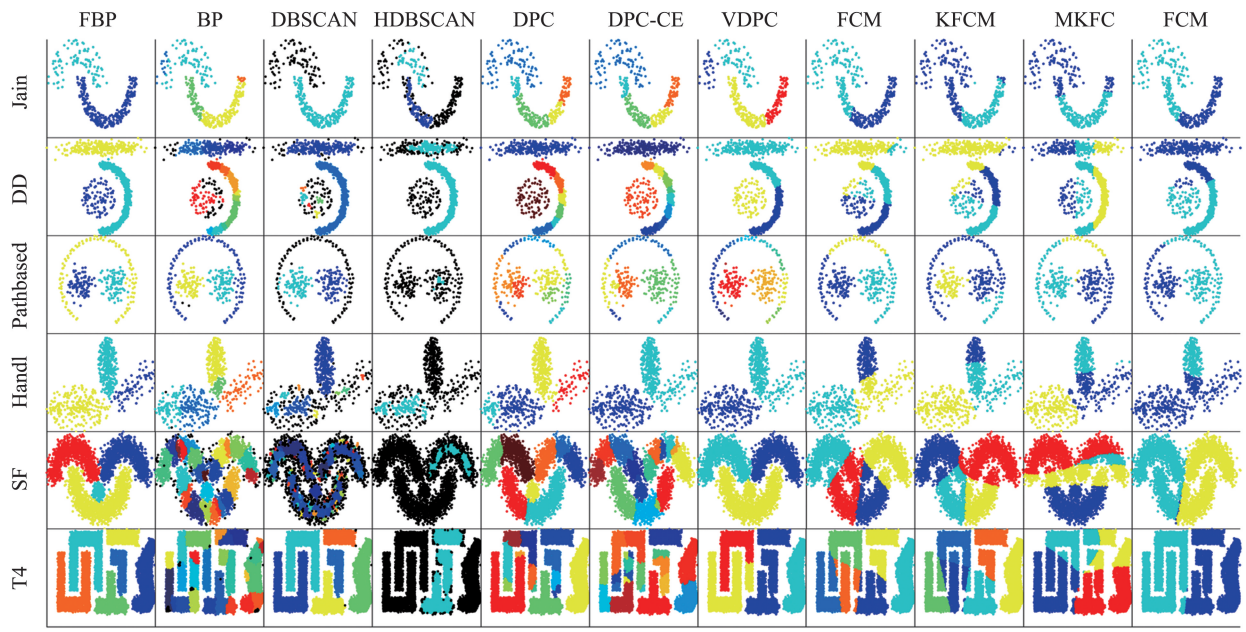

图 2　人工数据集上的聚类结果

Fig.2　Clustering results on synthetic datasets

DD 数据集具有 3 个形状各异的簇,其中球形簇密度最稀疏,长条形簇密度次之,月牙形簇密度最稠密。仅有 FBP 算法能准确发现该数据集的整体结构,BP 算法将 2 个较为稠密的簇划分为诸多更小的微簇,DBSCAN 将稀疏的圆形簇大部分识别为噪声,HDBSCAN 并未识别出稀疏的圆形簇,DPC 和 DPC-CE 将月牙形簇错分为多类,VDPC 将月牙形簇的一部分数据点错误地划分给长条形簇,FCM、KFCM、MKFC 和 PCM 均产生了错误的划分结果。

Jain 和 DD 数据集均为密度不均匀数据,FBP 算法能较好地处理密度不均匀的数据,这可能缘于本文所采用的密度估计方法。

Pathbased 数据集是一个环形簇包裹着 2 个高斯球形簇,这 3 个簇彼此紧挨,其中右侧簇和环形簇的右半部分存在重叠区域。FBP 和 BP 均取得了不错的聚类结果,但 BP 算法依然对几个处于边界区域的数据点产生了错误的分配,而 FBP 得到了完全正确的划分;DBSCAN 将环形簇识别为噪声;HDBSCAN 也将大部分数据点视为噪声数据;DPC、DPC-CE、VDPC、FCM、KFCM、MKFC 和 PCM 均无法检测出正确的簇结构。

Handl 数据集包含 3 个椭圆形簇,这 3 个簇不仅有着不同的密度分布形式,还存在着边界重叠的情况,因此,该数据集同时具有密度不均匀和边界模糊的特性。从图 1 中不难发现,仅有 FBP 算法给出了较为准确的聚类结果,BP 算法产生了过度划分的情况,DPC 准确检测出 2 个簇,DBSCAN、HDBSCAN、DPC-CE、VDPC、FCM、KFCM、MKFC 和 PCM 均产生了错误的划分结果。

Pathbased 和 Handl 为 2 组存在模糊边界问题的数据集,FBP 算法能较好地处理边界模糊的聚类划分问题,这缘于本文提出的模糊边界分配策略。

SF 数据集由 3 个月牙形簇和一个小型的椭圆形簇组成,3 个月牙形簇将椭圆形簇紧密的包裹在内,各簇彼此紧挨。该数据集存在簇彼此嵌套、缠绕的复杂分布形式。FBP 算法对于该复杂分布依然具有不错表现。BP 和 DPC 算法将 4 个簇错误地划分为诸多微型簇,其他算法也没有给出正确结果。

T4 数据集由 6 个形状各异的簇构成,每个簇都呈现不规则的形状,且彼此嵌套、缠绕,分布形式较为复杂。FBP 算法取得了最佳聚类效果,而其余对比算法均未完成该挑战。

SF 和 T4 为 2 组具有复杂分布的数据集,FBP 算法能发现分布不规则的簇,这缘于所采用的核心聚类划分方式和模糊边界分配策略。

为了定量地比较各算法在这些人工数据集上的性能,表 3 展示了 FBP 和 10 种对比算法在 ARI 和 NMI 指标上的表现,其中数字加粗意味着对应算法在该数据集上取得最高得分。FBP 算法在 6 组人工数据集的各个指标上均取得了最好的结果,其中在 Jain、Pathbased 和 T4 数据集上与真实标签完全匹配(得分为 1)。实验结果表明,FBP 算法可以识别具有不同密度、区域重叠和分布复杂的簇。

表 3 人工数据集上的性能评估

Table 3 Performance of the evaluated algorithms on synthetic datasets

Algorithm	Jain			DD		
	ARI	NMI	params	ARI	NMI	params
FBP	**1.000**	**1.000**	$k=19$	**0.992**	**0.980**	$k=24$
BP	0.478	0.677	$k=25$	0.154	0.530	$k=25$
DBSCAN	**1.000**	**1.000**	$\varepsilon=3.1$, MinPts=28	0.962	0.856	$\varepsilon=0.1$, MinPts=5
HDBSCAN	0.209	0.372	$k=5$, $m_c=5$	0.527	0.488	$k=28$, $m_c=20$
DPC	0.334	0.617	$d_c=6\%$, $\alpha=2.5$	0.256	0.619	$d_c=4\%$, $\alpha=2$
DPC-CE	0.304	0.609	$d_c=2\%$, $\alpha=2.5$	0.214	0.589	$d_c=4\%$, $\alpha=2$
VDPC	**1.000**	**1.000**	$d_c=5\%$, $\delta=5$	0.302	0.509	$d_c=3\%$, $\delta=1$
FCM	0.318	0.367	$M=1.5$	0.213	0.277	$M=4$
KFCM	0.239	0.323	$M=3.5$, $\delta=2$	0.204	0.299	$M=2$, $\delta=2$
MKFC	0.421	0.448	$M=1.5$	0.221	0.245	$M=3.5$
PCM	0.272	0.362	$M=3.5$	0.211	0.287	$M=2$
Algorithm	Pathbased			Handl		
	ARI	NMI	params	ARI	NMI	params
FBP	**1.000**	**1.000**	$k=9$	**0.988**	**0.974**	$k=30$
BP	0.980	0.969	$k=9$	0.659	0.785	$k=30$
DBSCAN	0.901	0.871	$\varepsilon=3.35$, MinPts=30	0.720	0.668	$\varepsilon=0.1$, MinPts=5
HDBSCAN	0.016	0.126	$k=5$, $m_c=20$	0.250	0.358	$k=9$, $m_c=15$
DPC	0.342	0.556	$d_c=2\%$, $\alpha=0.5$	0.844	0.828	$d_c=5\%$, $\alpha=1.5$
DPC-CE	0.520	0.611	$d_c=4\%$, $\alpha=1$	0.762	0.818	$d_c=8\%$, $\alpha=2.5$
VDPC	0.661	0.707	$d_c=2\%$, $\delta=2$	0.762	0.818	$d_c=5\%$, $\delta=1$
FCM	0.465	0.550	$M=1.5$	0.625	0.654	$M=1.5$
KFCM	0.454	0.545	$M=1.5$, $\delta=1$	0.560	0.672	$M=3$, $\delta=0.5$
MKFC	0.426	0.497	$M=4$	0.568	0.623	$M=2$
PCM	0.409	0.513	$M=4$	0.330	0.437	$M=4$
Algorithm	SF			T4		
	ARI	NMI	params	ARI	NMI	params
FBP	**0.978**	**0.953**	$k=23$	**1.000**	**1.000**	$k=26$
BP	0.201	0.599	$k=30$	0.361	0.737	$k=50$
DBSCAN	0.135	0.490	$\varepsilon=0.1$, MinPts=5	0.999	0.997	$\varepsilon=0.1$, MinPts=16
HDBSCAN	0.127	0.304	$k=21$, $m_c=20$	0.271	0.518	$k=46$, $m_c=20$
DPC	0.553	0.754	$d_c=4\%$, $\alpha=2.5$	0.828	0.883	$d_c=5\%$, $\alpha=2.5$
DPC-CE	0.346	0.662	$d_c=6\%$, $\alpha=2.5$	0.541	0.798	$d_c=5\%$, $\alpha=2.5$
VDPC	0.894	0.898	$d_c=2\%$, $\delta=2$	0.724	0.826	$d_c=2\%$, $\delta=1$
FCM	0.452	0.495	$M=1.5$	0.634	0.724	$M=4$
KFCM	0.542	0.560	$M=2.5$, $\delta=1.5$	0.644	0.734	$M=3$, $\delta=1.5$
MKFC	0.423	0.474	$M=4$	0.432	0.533	$M=4$
PCM	0.329	0.416	$M=4$	0.372	0.601	$M=2.5$

4.3 真实数据集结果

为了测试 FBP 算法在复杂真实环境下的表现,我们展示并分析了所有算法在 6 组真实数据集上的实验结果。表 4 给出了所有算法在 6 组数据集上的实验结果,其中数字加粗意味着对应算法在该数据集上取得最高得分。

表 4　真实数据集上的性能评估

Table 4　Performance of the evaluated algorithms on real-world datasets

Algorithm	Glass			Control		
	ARI	NMI	params	ARI	NMI	params
FBP	**0.693**	**0.769**	$k=6$	**0.680**	**0.856**	$k=19$
BP	0.393	0.670	$k=5$	0.626	0.805	$k=10$
DBSCAN	0.575	0.678	$\varepsilon=4.35$, MinPts=9	0.136	0.480	$\varepsilon=1.35$, MinPts=30
HDBSCAN	0.255	0.370	$k=5$, $m_c=20$	0.236	0.476	$k=13$, $m_c=20$
DPC	0.672	0.753	$d_c=10\%$, $\alpha=2.5$	0.540	0.741	$d_c=10\%$, $\alpha=2.5$
DPC-CE	0.642	0.765	$d_c=8\%$, $\alpha=2$	0.543	0.744	$d_c=9\%$, $\alpha=2.5$
VDPC	0.363	0.713	$d_c=5\%$, $\delta=5$	0.554	0.681	$d_c=10\%$, $\delta=1$
FCM	0.570	0.762	$M=2$	0.619	0.732	$M=1.5$
KFCM	0.201	0.467	$M=1.5$, $\delta=2$	0.660	0.754	$M=1.5$, $\delta=2$
MKFC	0.242	0.344	$M=1.5$	0.646	0.763	$M=1.5$
PCM	0.282	0.464	$M=2$	0.536	0.686	$M=1.5$
Algorithm	Dig			PageB		
	ARI	NMI	params	ARI	NMI	params
FBP	**0.795**	**0.858**	$k=9$	0.413	**0.305**	$k=20$
BP	0.397	0.698	$k=10$	0.095	0.210	$k=23$
DBSCAN	0.557	0.741	$\varepsilon=1.35$, MinPts=4	**0.434**	0.248	$\varepsilon=0.1$, MinPts=50
HDBSCAN	0.355	0.675	$k=2$, $m_c=5$	0.318	0.130	$k=29$, $m_c=20$
DPC	0.781	0.849	$d_c=2\%$, $\alpha=2.5$	0.335	0.213	$d_c=8\%$, $\alpha=1$
DPC-CE	0.714	0.826	$d_c=4\%$, $\alpha=2.5$	0.384	0.255	$d_c=8\%$, $\alpha=1$
VDPC	0.180	0.597	$d_c=2\%$, $\delta=2$	0.012	0.050	$d_c=10\%$, $\delta=1$
FCM	0.251	0.464	$M=1.5$	0.091	0.142	$M=1.5$
KFCM	0.452	0.559	$M=2.5$, $\delta=1$	0.076	0.147	$M=1.5$, $\delta=1.5$
MKFC	0.000	0.000	$M=1.5$	0.049	0.090	$M=1.5$
PCM	0.254	0.432	$M=2.5$	0.036	0.052	$M=1.5$
Algorithm	Optdigits			Crowd		
	ARI	NMI	params	ARI	NMI	params
FBP	**0.832**	**0.877**	$k=12$	0.425	0.403	$k=46$
BP	0.095	0.210	$k=23$	0.354	0.349	$k=46$
DBSCAN	0.532	0.732	$\varepsilon=1.35$, MinPts=23	0.263	0.339	$\varepsilon=0.35$, MinPts=35
HDBSCAN	0.409	0.653	$k=2$, $m_c=5$	0.285	0.347	$k=41$, $m_c=5$
DPC	0.736	0.808	$d_c=2\%$, $\alpha=2.5$	0.006	0.365	$d_c=3\%$, $\alpha=1$
DPC-CE	0.723	0.812	$d_c=4\%$, $\alpha=2.5$	0.031	0.371	$d_c=2\%$, $\alpha=1$
VDPC	0.141	0.576	$d_c=2\%$, $\delta=2$	0.000	0.000	$d_c=2\%$, $\delta=2$
FCM	0.249	0.408	$M=3.5$	0.106	0.242	$M=1.5$
KFCM	0.356	0.528	$M=2.5$, $\delta=1$	0.121	0.246	$M=1.5$, $\delta=2$
MKFC	0.000	0.000	$M=1.5$	0.174	0.258	$M=1.5$
PCM	0.246	0.414	$M=2$	0.171	0.114	$M=1.5$

在 Glass 数据集中,FBP 算法在 ARI 指标上高出原始 BP 算法 30%;在 NMI 指标上高出原始 BP 算法 10%以上;相较于 10 种对比算法,FBP 算法在这 2 个指标上平均提升了 31%和 22%。在 Control 数据集中,FBP 算法在 ARI 指标上取得 0.68,BP 算法为 0.626,高出 5%;FBP 算法在 NMI 指标上取得 0.856,而第二名 BP 算法在该指标上的得分为 0.805,高出 5%;相较于 10 种对比算法,FBP 算法在这 2 个指标上平均都提升了 17%。在 Dig 数据集中,FBP 算法在 ARI 和 NMI 指标上分别高出 BP 算法 40%和 16%;相较于 10 种对比算法,FBP 算法在这 2 个指标上平均提升了 43%和 25%。在 PageB 数据集中,FBP 算法在 ARI 指标上的得分为 0.431,而第二名 BP 算法的 ARI 得分为 0.095,在该指标上高出一个数量级;在 NMI 指标上本算法高出原始 BP 算法 10%左右;相较于 10 种对比算法,本算法在这 2 个指标上平均提升了 21%和 13%。在 Opdis-

gits 数据集中,FBP 算法在 ARI 和 NMI 上分别取得了 0.832 和 0.877 的得分,而 BP 算法在这 2 个指标上的得分仅为 0.095 和 0.210;相较于 10 种对比算法,本算法在这 2 个指标上平均提升了 44%和 32%。在 Crowd 数据集中,FBP 算法在 2 个指标上也高出 BP 算法 5%以上;相较于 10 种对比算法,本算法在这 2 个指标上平均提升了 25%和 12%。本算法在 6 组真实数据集上的所有指标均优于绝大多数对比算法。

综上所述,FBP 算法在处理复杂真实数据时,依然取得令人满意的表现。

5 结束语

本文提出了一种基于密度的聚类算法,名为 FBP 算法。FBP 算法可以处理具有密度可变、边界模糊、分布复杂的数据。具体而言,本文采用了一种基于 Cauchy 核的密度估计方式来计算数据点密度;接着,使用边界剥离策略来区分核心数据和边界数据;而后,使用类 DBSCAN 算法进行核心数据点的聚类;最后,采用所提出的模糊分配策略实现边界数据的划分。在人工和真实数据集上的实验结果表明,本文提出的 FBP 算法相比于对比算法具有更好的表现,尤其是在处理密度不均匀、边界模糊等复杂分布的数据时,具有明显的优势。

后续工作中,将尝试开发一种自动确定近邻数目的机制,使得模糊边界剥离聚类算法在无需任何参数的情况下完成聚类工作。

参考文献:

[1] 徐晓,丁世飞,丁玲. 密度峰值聚类算法研究进展[J]. 软件学报,2022,33(5):1800-1816.
XU Xiao, DING Shifei, DING Ling. Survey on density peaks clustering algorithm[J]. Journal of Software, 2022, 33(5): 1800-1816.

[2] PENG D, GUI Z, WANG D, et al. Clustering by measuring local direction centrality for data with heterogeneous density and weak connectivity[J]. Nature Communications, 2022, 13(1):5455.

[3] 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 Second International Conference on Knowledge Discovery and Data Mining. Menlo Park:AAAI Press, 1996:226-231.

[4] CAMPELLO R J, MOULAVI D, SANDER J. Density-based clustering based on hierarchical density estimates[C]//Proceedings of the 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining. Berlin:Springer, 2013:160-172.

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

[6] CHEN H, LIANG M, LIU W, et al. An approach to boundary detection for 3D point clouds based on dbscan clustering[J]. Pattern Recognition, 2022, 124:108431.

[7] OUYANG T, PEDRYCZ W, PIZZI N J. Rule-based modeling with dbscan-based information granules[J]. IEEE Transactions on Cybernetics, 2021, 51(7):3653-3663.

[8] LU J, ZHAO Y, TAN K L, et al. Distributed density peaks clustering revisited[J]. IEEE Transactions on Knowledge and Data Engineering, 2022, 34(8):3714-3726.

[9] RASOOL Z, ZHOU R, CHEN L, et al. Index-based solutions for efficient density peak clustering[J]. IEEE Transactions on Knowledge and Data Engineering, 2022, 34(5):2212-2226.

[10] 盛锦超,杜明晶,李宇蕊,等. 结合柯西核的分类型数据密度峰值聚类算法[J]. 计算机工程与应用, 2022, 58(18):162-171.
SHENG Jingchao, DU Mingjing, LI Yurui, et al. Cauchy kernel-based density peaks clustering algorithm for categorical data [J]. Computer Engineering and Applications, 2022, 58(18):162-171.

[11] AVERBUCH-ELOR H, BAR N, COHEN-OR D. Border-peeling clustering[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020, 42(7):1791-1797.

[12] ANKERST M, BREUNIG M M, KRIEGEL H P, et al. Optics:ordering points to identify the clustering structure[C]//Proceedings of the 1999 International Conference on Management of Data. New York: ACM, 1999:49-60.

[13] GUO W, WANG W, ZHAO S, et al. Density peak clustering with connectivity estimation[J]. Knowledge-Based Systems, 2022, 243:108501.

[14] WANG Y, WANG D, ZHOU Y, et al. VDPC:variational density peak clustering algorithm[J]. Information Sciences, 2023, 621:627-651.

(下转第 50 页)