ODIC-DBSCAN: 一种新的簇内孤立点分析算法

王跃飞[1] 于 炯[1,2] 苏国平[3] 钱育蓉[2] 廖 彬[4] 刘 粟[1]

摘 要 长期以来, 孤立点的检测一直聚焦于簇边缘的离散点, 当聚类后簇的数目低于实际数目, 或孤立点被伪装在簇内的情况下, 簇内孤立点的判定则会更加困难. 为判定簇内孤立点, 提出一种基于密度聚类 DBSCAN (Density based spatial clustering of application with noise) 的簇内孤立点检测方法 ODIC-DBSCAN (Outlier detection of inner-cluster based on DBSCAN). 首先在建立距离矩阵的基础上, 通过半径获取策略得到针对该点集的 k 个有效半径 *Radius* 集合, 并据此构造密度矩阵; 然后建立点集覆盖模型, 提出了相邻有效半径构造的覆盖多维体能够覆盖点集的思想, 并通过拉格朗日乘子法求取最优的覆盖多维体数目之比, 输出点比阈值组; 最后重建 ODIC-DBSCAN 的孤立点检测方法, 以簇发生融合现象作为算法终止的判定条件. 实验通过模拟数据集, 公开 benchmark 与 UCI 数据集共同验证了 ODIC-DBSCAN 算法, 展示了聚类过程; 分析了算法性能; 并通过与其他聚类、孤立点判定方法的对比, 验证了算法对簇内孤立点的判定效果.

关键词 聚类, DBSCAN, 簇内孤立点, 密度关联, 孤立点检测

引用格式 王跃飞, 于炯, 苏国平, 钱育蓉, 廖彬, 刘粟. ODIC-DBSCAN: 一种新的簇内孤立点分析算法. 自动化学报, 2019, **45**(11): 2107−2127

DOI 10.16383/j.aas.c180617

ODIC-DBSCAN: A New Analytical Algorithm for Inliers

WANG Yue-Fei[1] YU Jiong[1,2] SU Guo-Ping[3] QIAN Yu-Rong[2] LIAO Bin[4] LIU Su[1]

Abstract Outlier detection has been focused on the discrete points of cluster edges for a long time. When the number of clusters is less than the actual number, or the outliers are disguised within the cluster, the detection of inliers becomes more difficult. Therefore, a new analytical algorithm for inliers ODIC-DBSCAN (Outlier detection of inner-cluster based on DBSCAN), which is based on DBSCAN (Density based spatial clustering of application with noise), is proposed. First, on the basis of establishing the distance matrix, the set of k effective radii for the set of points is obtained through the proposed Radius Obtaining Strategy, and the density matrix is constructed accordingly. Then, the point-set covering model is established and the idea that the covering multidimensional cube with adjacent effective radius can cover the point sets is proposed. The Lagrange multiplier method is used to obtain the optimal ratio of the number of covering multidimensional cubes, and the group of point ratio thresholds is obtained. Finally, the method of outlier detection based on ODIC-DBSCAN is reconstructed, and the fusion phenomenon of the clusters is taken as the terminating condition of the algorithm. The experiment verifies the ODIC-DBSCAN algorithm through three kinds of point sets: the synthetic point sets, the public clustering benchmarks and the UCI real-world datasets; the clustering process is demonstrated and the performances are analyzed. Besides, experimental results show that comparing to other clustering and outlier detection methods, the ODIC-DBSCAN algorithm is able to determine the inliers more effectively.

Key words Clustering, DBSCAN, inliers, density correlation, outlier detection

Citation Wang Yue-Fei, Yu Jiong, Su Guo-Ping, Qian Yu-Rong, Liao Bin, Liu Su. ODIC-DBSCAN: a new analytical algorithm for inliers. *Acta Automatica Sinica*, 2019, **45**(11): 2107−2127

聚类算法在计算机科学、交通运输、电子商务[1−3] 等越来越多的学科、领域中均有重要应用. 特别在计算机领域, 聚类算法是机器学习、数据挖掘及人工智能的基础. 相应的, 各类丰富的聚类思想及其衍生算法应运而生. 总体上, 聚类可分为基于划分、密度、网格、层次等方法, 不同的方法采用了不同的聚类思想和技术, 致使聚类结果具备多样化特征.

聚类技术同样是孤立点检测的重要手段. 孤立

收稿日期 2018-09-15 录用日期 2019-03-29
Manuscript received September 15, 2018; accepted March 29, 2019
国家自然科学基金 (61862060, 61462079, 61562086, 61562078) 资助
Supported by National Natural Science Foundation of China (61862060, 61462079, 61562086, 61562078)
本文责任编委 陈德旺
Recommended by Associate Editor CHEN De-Wang
1. 新疆大学信息科学与工程学院 乌鲁木齐 830046 2. 新疆大学软件学院 乌鲁木齐 830008 3. 新疆维吾尔自治区经济和信息化委员会 乌鲁木齐 830000 4. 新疆财经大学统计与信息学院 乌鲁木齐 830012
1. College of Information Science and Engineering, Xinjiang University, Urumqi 830046 2. School of Software, Xinjiang University, Urumqi 830008 3. The Economic and Information Commission of Xinjiang Uyghur Autonomous Region, Urumqi 830000 4. School of Statistics and Information, Xinjiang University of Finance and Economics, Urumqi 830012

点概念的提出[4] 不仅使各行业的数据纯度得以有效保障, 同样为新知识的探索和挖掘提供了重要依据.

在目前的研究进展中, 有基于统计学[5−6], 基于距离[7−8], 基于密度[9−10] 与基于聚类等不同的检测思想, 在聚类技术的检测手段中, 一般是将生成的簇外点置为孤立点.

然而, 主流方法一般聚焦于簇间的孤立点检测, 而忽略了簇内的孤立点, 聚类算法更是将簇范围下的点均默认为正常点, 不启用二次审查. 实际上, 簇内的孤立点是有可能存在的, 主要原因是聚类属于无监督学习 (Unsupervised learning), 在没有指示标签的情况下, 不同的聚类方法可能产生多种结果, 包括簇的数目不同; 另一方面, 在大部分行业中孤立点是动态生成的, 如在医学领域中, 病变早期细胞组织的位置由正常区域逐渐位移, 使聚类的几何中心产生偏移, 因此及早地发现簇内异常对这类现象具有重大意义.

图 1 的 3 种现象描述了簇内孤立点的产生原因.

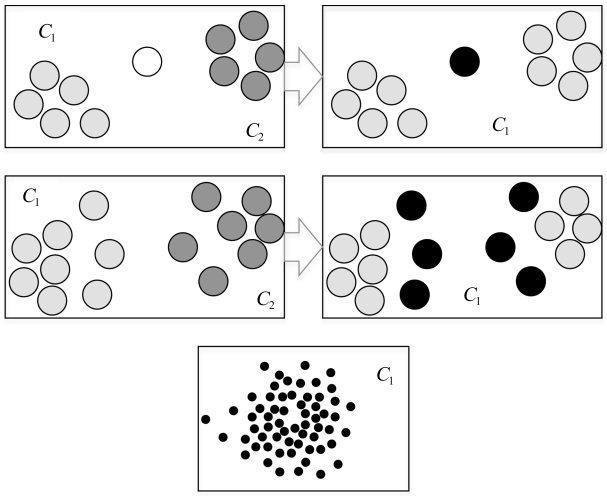


图 1 簇内孤立点产出原因

Fig. 1 The cause of inliers

现象 1: 当使用不同方法或参数, 使原先多个簇发生融合时, 先前在多簇间被判定为孤立点的对象会被判定为正常点, 产生误判.

现象 2: 当多个簇发生融合时, 若汇聚的一面为簇的边缘, 则汇聚后簇存在松散域, 其中的点可能具有孤立性质.

现象 3: 当点集紧密投影在区域内部时, 若有一处存在明显稀疏或空白, 则该区域下的点需要重点考证研究.

以上三类典型现象不仅描述了具有孤立性质的点被判定为正常点的情况, 同时可以将簇内孤立点按产生原因分为两类: 1) 聚类技术中, 不同的参数使簇发生融合时产生; 2) 数据的原始分布造成. 以上误判的出现影响了聚类效果, 污染了数据纯度, 降低了数据质量. 在聚类计算中, 忽略簇内孤立点容易造成簇中心位置的误判; 在各类生产生活以及应用中, 由于参数或方法的不当使用, 孤立点的忽略更可能会带来安全隐患. 为避免上述现象的出现, 需要对簇内孤立点进行更加深入的研究, 设计更为有效的簇内孤立点判定算法. 该算法需具有以下 3 方面的功能:

1) 能够分析簇间孤立点;

2) 能够分析簇内孤立点;

3) 不影响聚类效果且时间复杂度较低.

本文提出一种对簇内孤立点敏感的方法: ODIC-DBSCAN, 该方法考虑到不同半径下密度间的关系, 通过有效相邻半径构成的多维体能够覆盖点集的思想, 计算在不同密度下的相关参数, 重构了 DBSCAN 框架, 查找出簇内孤立点. 本文的主要工作有以下几方面:

1) 获取有效半径. 为检测簇内部的孤立点, 首先对数据集的密度进行层次划分, 并提出半径获取策略. 该方法能够根据点与点间的距离关系对任意数据集提取有效半径.

2) 提出点集覆盖的思想. 该思想以有效半径为基础, 分割覆盖为原则, 表示为相邻有效半径能将数据集完整覆盖. 在此基础上, 通过相邻有效半径构造出多条无差别曲线, 每一条曲线的点均能描述数据集被覆盖的情况, 并使用拉格朗日乘子法求取了最优值.

3) 从两方面重构了 DBSCAN 算法. 第一, 修改了 DBSCAN 中核心对象的定义, 以相邻半径间点数的比值代替了半径下点的数目. 第二, 提出了算法的终止条件, 以簇间是否发生融合现象作为聚类效果的依据.

本文第 1 节简要概述相关工作. 第 2 节首先介绍了 DBSCAN 的缺陷, 提出了解决方法; 并在后续的小节中详细介绍了 ODIC-DBSCAN 的模型研究, 依序包括半径获取策略 (第 2.2 节) 与点集覆盖模型 (第 2.3 节), 并在第 2.4 节中展开聚类算法的重构工作; 最后分析了算法时间复杂度. 实验部分在第 3 节中展开, 该部分展示了 ODIC-DBSCAN 算法的运行细节; 分析了算法的两项性能指标; 验证了算法对簇内孤立点辨别的优势, 并与其他同类算法的运行效果进行了对比.

1 相关工作

当前大部分聚类算法对簇内或簇间的孤立点检测均有一定的效果, 但传统算法由于关注聚类的思想、结果、效率、以及孤立程度等问题, 并未专注于簇内孤立点的发现, 因此对簇内孤立点并不敏

感. 总体上, 孤立点的研究是一个宽广的领域, 针对各个类型的数据, 孤立点检测有不同的方法[11], 如高维数据[12]、不确定数据[13]、流式数据[14]、网络数据[15]. 目前在基础研究中, 孤立点的分析与进展有以下方面: 文献 [9] 在密度聚类的基础上, 提出了一种将点数据分为多个层次的思想, 使同一层的数据在全局具有相似的特性, ISDBSCAN (Influence space DBSCAN) 思想与距离有关, 算法在目标对象的一定范围内根据邻居点距离的总和使目标对象分层. Duan 等[16] 针对密度聚类, 对个别小规模的簇被判定为孤立点的现象提出优化方法, 所提出的 LDBSCAN (Local-density based spatial clustering algorithm with noise) 方法能达到分离簇与点的目的; 文献 [16] 认为, 孤立点的现象不应仅仅是单独点离群的状态, 在较小的簇偏离点集主体区域时, 该簇内的点均应被认作孤立点; 在基于簇的孤立点 (Cluster-based outlier) 检测中, 文章提出了 LDBSCAN 算法, 并提出了离群簇孤立程度的计算方式. 文献 [17] 针对传统孤立点检测在大规模数据集中效果差强人意的现象, 提出了一种基于统计的孤立点检测方法. 首先通过三倍标准差法记录密度峰值, 其次将剩余数据以近邻原则指定到相应的簇中, 最后使用切比雪夫不等式与密度可达性来判定目标点是否为孤立点. 文献 [18] 使用分治思想 (Divide-and-conquer strategy) 提出一种孤立点识别策略: 初始时数据集被分为两个簇, 然后算法在全局过程中通过迭代来检查两个密度波峰是否密度可达, 以验证二者是否为同一个簇, 循环后最终将点指定到所属簇中. 上述工作计算了边缘密度, 因此当点的边缘密度低于阈值时被认作孤立点. 文献 [19] 提出了不再将孤立点定义为二值属性, 而是根据被孤立的程度给予不同的局部异常因子, 对于模糊点, 给定了 LOF (Local outlier factor) 的值边界. 在基于非二值属性的宏观思想下, 面向角度[20], 面向距离[7] 等检测方法应运而生. 以上方法能对任意数据对象赋予孤立程度 (Outlier-ness), 只是检测的思想不同. 文献 [20] 考虑角度因素, 并论证了角度特征对高维空间下的数据衡量更加精准; F-ABOD 方法比对任意三点构成的两个向量间夹角大小, 通过角度确定点距簇心的远近. 文献 [7] 着重距离因素, 对数据点建立双层邻居系统 (Neighborhood system), 并通过双层邻居的距离关系反馈核心点的孤立性.

聚类是一种检测孤立点的有效手段, 基于密度的检测思想是检验点间关系的有效方法. 在这些方法中, Rodriguez 等提出一种基于密度峰值聚类的方法 DPC (Density peaks clustering, DPC)[21], 该方法认为, 簇中心的密度一定比边缘处高, 且与距它密度更高的簇心距离较大; 在此思想上, 算法不断寻找被低密区域分离的高密区域. 该方法的优势是不需迭代, 因此运行时间短. 在此基础上, 相关的研究工作陆续展开, 包括参数优化[22], 聚类的集成[23], 自适应模糊聚类[24], 以及社交网络下的聚类应用[25] 等. 另一种基于密度的方法是通过数据点间执行 “信息传递” 而进行聚类[26]. AP (Affinity propagation) 算法不需指定簇的数目, 而是通过建立吸引信息 (Responsibility) 与归属信息 (Availability) 两类矩阵, 并引入衰减系数对两类信息进行衰减迭代, 当矩阵值稳定时, 聚类结束. 在 AP 技术的研究基础上, 一些相关的工作也被相继提出, 如图像中半监督的 AP 聚类方法[27], 基于层次划分的 AP 聚类[28] 等. OPTICS (Ordering points to identify the clustering structure) 是将空间数据依据密度执行聚类的一种方法[29], 在 DBSCAN 的基础上, OPTICS 的聚类对象能应用于多密度点集, 但算法生成的可达距离 RD (Reachability distance) 不能显示的生成聚类结果, 仅生成包含聚类信息的有序对象列表. 另外, 针对一种密度阈值无法满足全局数据分布的需要, 文献 [30] 提出一种基于密度比 (Density-ratio) 的思想, 并将其应用于 DBSCAN、OPTICS 等方法. 该思想包含 Reconditioning approach 与 Rescaling approach 两类方法, 前者提出使用密度比而非密度阈值的解决办法, 可以植入相关的密度聚类; 后者对数据集的坐标进行重投射, 使生成的范围点集密度接近预设的密度比参数, 这样, 使原数据集的密度标准化之后再进行聚类. 另外, 在算法的应用研究中, 孤立点的识别在不同领域中已经有了较大程度的进展. 在关于图像的孤立点判定识别中, 文献 [31] 通过计算簇内图像数据偏离的评分确定孤立程度, 这个评分根据将簇内点抽离后图像信息的变化量来确定. 文献 [32] 通过使用随机森林与 DBSCAN 聚类方法对类似姓名等文本做出识别处理. 在无线传感器网络的孤立点检测中, 借用 DBSCAN 方法, 在计算参数, 空间时态数据库下识别出具体的类信息, 且能够识别出离群点[33].

2 模型与策略

2.1 问题分析

基于密度的聚类 (Density-based clustering) 能根据样本间的密度关系确定簇的形成, 通过邻域大小与该区域内点数目的关系考察点之间的连通性, 刻画簇的构成. DBSCAN 算法能将接收到的核心对象 (Core object) 按不同密度, 以不同形状执行聚类, 但忽略了点集不同密度间的联系. 1) DBSCAN 算法仅能接受一种密度条件, 并且在没有先验知识的条件下, 半径无法明确指定. 2) 局部数据并非遵

循高斯分布, 在图 1 中的分布现象中, 由于输入参数的限制, DBSCAN 不能很好地处理离散点的分析工作. 3) 密度聚类错过了微观角度下点与点之间的关系. 在一个簇中, 虽然宏观角度下呈现高密特征, 但微观下个别特殊的点与点间可能包含差异较大的距离, 遗憾的是, DBSCAN 不能识别类似的特殊点. 在簇扩充的工作中, DBSCAN 通过 *epsilon* 邻域中点的个数确定该点是否为核心对象; 该方法能够计算目标点的密度状态, 但忽略了不同密度条件下的不同结果.

簇内孤立点的特性在于目标点与周围的邻居具有密度差异, 该差异可能由点的分布直接造成, 也可能由聚类算法的参数使用不当生成. 在点集中, 由于其与周围邻居密度差异较小而经常被忽视, 因而簇内孤立点的检验需要算法对微小的密度差异具备敏感性.

定义 1. 簇内孤立点 (Inlier). 表示在数据集下的任意簇中, 当前目标点的空间密度较邻居点更稀疏的一类对象. 由数据的原始分布, 或不当参数造成簇融合时算法对簇间点的误判两类原因产生.

当簇发生融合时, 簇内与簇间孤立点容易被算法错误判定. 如图 1 所展示的孤立点与正常点相互转换的情况. 为解决以上问题, 算法改进的核心是点在不同半径下的密度关系, 因此本文关注于点在不同范围下的密度关联, 并重定义 DBSCAN 的核心对象. 具体技术是根据点集的分布情况提取若干由小到大的有效半径, 在密度聚类中根据该点在两个半径下密度之比是否大于数据预处理输出的阈值来判定该点是否属于核心对象.

如图 2 所示, 左部为 DBSCAN 算法的核心对象确定方法. 当在 *epsilon* 半径下, 覆盖面下的点数目超过 $MinPts$ 时, 该点 (图中为圆心黑色点) 为核心对象. 右部为 ODIC-DBSCAN 的确定方法: 通过前期数据预处理输出的典型半径 r_i、r_j, 确定两个半径下覆盖面积的密度比值, 通过比值是否大于前期数据预处理输出的阈值来确定该点是否为核心对象.

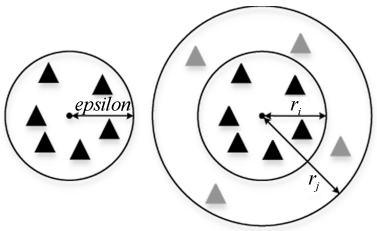


图 2 DBSCAN 与 ODIC-DBSCAN 的核心对象确定方法

Fig. 2 The core object confirmation method of DBSCAN and ODIC-DBSCAN

通过上述方法, 能够确定目标点在有效的、逐渐扩大的两个范围内相邻数据量的变化情况. 在孤立点分析中, ODIC-DBSCAN 算法循环地输入有效半径集合 *Radius* 内由小到大相邻半径构成的开圆密度比, 并将此作为聚类核心对象判定的阈值. 当真实情况超过该阈值, 表示该点的范围密度较高, 为簇内点; 反之该点周围密度较低, 为偏边缘点. 当选择 *Radius* 的半径由小至大时 (如图 3 所示), 簇内孤立点的判定逐渐变得粗略, 容易产生融合现象, 这表示阈值过于宽泛; 为避免融合, 应选择簇发生融合之前的结果作为聚类结果.

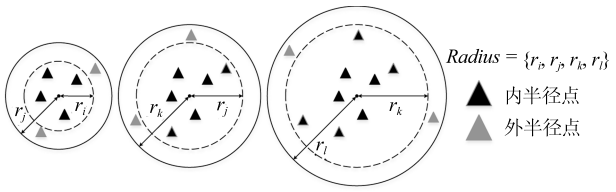


图 3 ODIC-DBSCAN 核心对象在半径集合 *Radius* 下的遴选

Fig. 3 The selection of ODIC-DBSCAN core objects from *Radius* set

上述思想能够有效判断以目标点为中心, 在多重半径范围内的数据点密度的变化情况, 并以此作为依据推断该点的位置, 确定该点是否具有孤立性质.

ODIC-DBSCAN 算法包含两个参数, 分别位于算法流程下数据预处理、簇内孤立点分析这两个部分. 如图 4 所示, 在数据预处理部分, 首先根据数据集的距离分布状况, 提出一种半径获取策略, 该策略能够将数据集的关键距离遴选出作为半径集合 *Radius*, 并构建密度矩阵. 当获取到 *Radius* 后, 提出一种思想, 证明任意点集能够由集合 *Radius* 构成的覆盖多维体完全覆盖; 基于此思想, 使 *Radius* 内相邻半径 r_i, r_j 构成的覆盖多维体覆盖点集, 并计算对应半径构成覆盖多维体的数目, 最终构造出点比阈值组. 在 ODIC-DBSCAN 算法处理部分, 重新定义了 DBSCAN 中的核心对象, 并提供了一种孤立点分析检测的方法. 在以上两部分顺序进行后, 算法最终输出簇内的孤立点并生成图像.

2.2 半径获取策略

定义 2. 距离矩阵. 描述了点集 P 中任意两点之间的欧氏距离.

$$\boldsymbol{dist}_{m\times m}=\begin{bmatrix} 0 & d_{\langle 1,2\rangle} & d_{\langle 1,3\rangle} & \cdots & d_{\langle 1,m\rangle} \\ & 0 & d_{\langle 2,3\rangle} & \cdots & d_{\langle 2,m\rangle} \\ & & 0 & \cdots & d_{\langle 3,m\rangle} \\ & & & \ddots & \vdots \\ & & & & 0 \end{bmatrix} \tag{1}$$

设点集 $P_{\langle d,m\rangle}=\bigcup_{i=1}^{m}x_i$ 中点的维度为 d, 共计 m 个点. 则该点集的距离矩阵为 $m\times m$. 式 (1) 展示了 ***dist*** 的基本度量形式, 该矩阵为非负实数构成的对称阵, 且主对角线元素为 0.

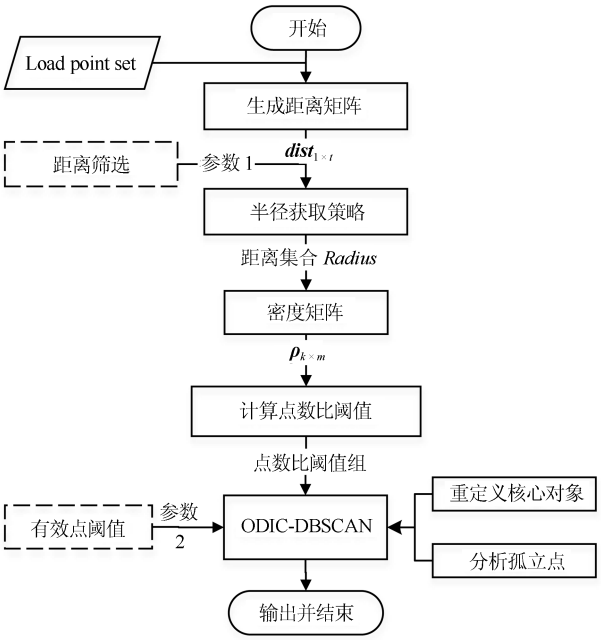


图 4　ODIC-DBSCAN 处理流程

Fig. 4　The processing flow of ODIC-DBSCAN

$\boldsymbol{dist}_{m\times m}$ 是距离矩阵的表现形式, 在实际操作中, 需要将其转化为一行多列的矩阵格式. 因此距离矩阵也被记为 $\boldsymbol{dist}_{1\times t}$, 其中 t 为式 (1) 中对角线以上距离元素的个数.

定义 3. 密度矩阵. 密度矩阵宏观统计点集 P 中每个点在一定范围内的点数量与面积之比.

$$\boldsymbol{\rho}_{k\times m}=\begin{bmatrix}\rho_{\langle x_1,r_1\rangle} & \rho_{\langle x_2,r_1\rangle} & \cdots & \rho_{\langle x_m,r_1\rangle}\\ \rho_{\langle x_1,r_2\rangle} & \rho_{\langle x_2,r_2\rangle} & \cdots & \rho_{\langle x_m,r_2\rangle}\\ \vdots & \vdots & \ddots & \vdots\\ \rho_{\langle x_1,r_k\rangle} & \rho_{\langle x_2,r_k\rangle} & \cdots & \rho_{\langle x_m,r_k\rangle}\end{bmatrix}\tag{2}$$

式 (2) 的密度矩阵中, 列数 m 代表数据集的点对象, 行表示根据点集的分布而规划的 k 个半径距离. 具体地说, 在矩阵元素 $\rho_{\langle x_m,r_k\rangle}$ 中, x_m 表示点对象, r_k 表示该研究点的密度度量是在以半径为 r_k 下的开圆中, 其中 $r_1<r_2<\cdots<r_k$, $Radius=\{r_1,r_2,\cdots,r_k\}$. 式 (2) 的核心问题在于, 如何选择合适的有效半径构成 $Radius$ 集合.

在数据集的距离矩阵内, 所有距离元素在长度范围下的数目分布中, 其波峰一般不少于一个. 这表明, 多个波峰与波谷的存在形式使距离矩阵的数据密集程度具有较强不确定性, 不能简单地通过数据边缘或中心分布确定距离集合具备正态分布、卡方分布、t 分布等分布曲线形态. 为有效确定数据集内点与点间的距离分布情况, 提出半径获取策略 (Radius obtaining strategy).

该策略的主要思想是: 自顶向下, 逐渐细化分割, 删去距离为空的范围.

如图 5～6 所示, 在数据分布中有簇 A 与簇 B, 共计距离 45 个. 其中簇内距离 21 个 (实线), 簇间距离 24 个 (虚线), 一般地, 簇内距离小于簇间距离. 设置一条标准线作为基准, 将距离元素的分布划分为稠密处与稀疏处. 稠密处为波峰 1、2, 分别汇聚了簇内点的相互距离元素, 以及簇间的距离元素; 波谷 1 处存在极少该范围内的距离. 取出波峰 1, 执行同样步骤, 划分若干区间后重新制定新的标准, 将波峰 1 划分为子波谷 1、子波峰 1、子波谷 2; 持续以上工作, 直至波峰波谷细化完毕, 最终确定若干区间, 并确定区间内平均值的大小.

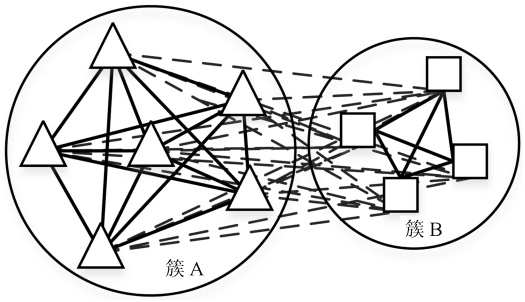


图 5　点集内的距离关系

Fig. 5　The relationship of distance within the point set

算法 1. 半径获取策略

输入: 距离矩阵 $\boldsymbol{dist}_{1\times t}$.

输出: 有效半径集合 $Radius$.

步骤 1. 数据预处理. 加载数据集, 并置入集合 $OriginalDist$. 在获取原始数据的欧氏距离后可将向量转化为式 (1) 的距离矩阵.

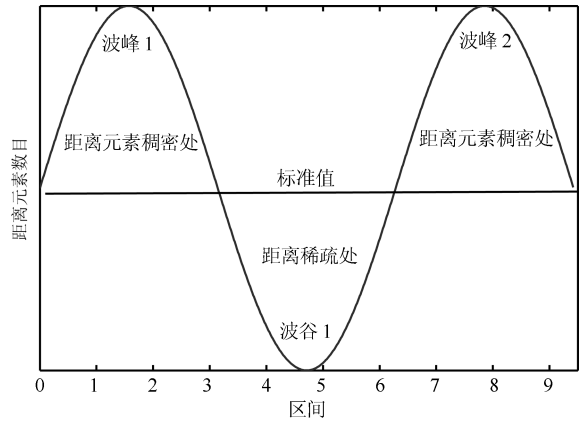
步骤 2. 算法背景说明. 设 $OriginalDist$ 最大元素 $MaxDist$, 集合元素数目 num, 区间 (Range) 数目 $RangeNum$, 则步长 $StepSize=MaxDist/RangeNum$. 制定一个标准 $StandardScore$, 该标准表示每个区间平均应具有多少个距离元素. 如 $MaxDist=100$, 若区间数目为 5, 则以步长 20 为单位, 划分区间分别是 (0, 20], (20, 40], (40, 60], (60, 80], (80, 100], 若共有距离数目为 100 个, 则 $StandardScore$ 为 20.

步骤 3. 区间统计. 遍历集合, 逐一统计区间内元素的数目.

步骤 4. 核心计算. 在扫描区间时, 1) 遴选区间; 2) 确定波峰波谷线. 当扫描区间内无元素时, 直接删除该元素数目为零的区间. 当扫描区间内包含元

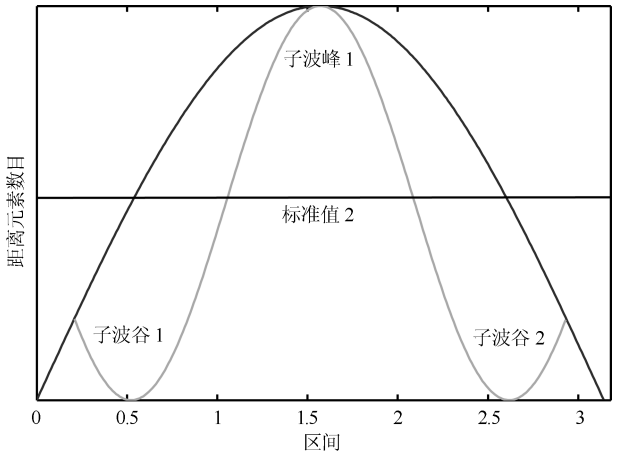
素时, 逐一计算区间的得分:

$$score = \frac{num_{(lower,upper)}}{RangeNum} \tag{3}$$



(a) 首次距离元素分布划分

(a) First division of distance distribution



(b) 距离元素分布再次划分

(b) Division of distance distribution once again

图 6 距离元素分布的划分过程

Fig. 6 The division process of distance distribution

若得分高于标准值 *StandardScore*, 则记录并扫描下一区间, 直至遇到低于标准值区间 (或元素为零区间) 而终结. 合并以上区间为一大区间 (Combined range), 确认为波峰 (波谷), 并计算、标记该大区间得分 *score*.

定义一个得分矩阵 ***Score***, 该矩阵接收核心方法的返回结果.

$$\boldsymbol{Score} = \begin{bmatrix} SEQ_1 & lb_1 & ub_1 & bool_1 & score_1 \\ SEQ_2 & lb_2 & ub_2 & bool_2 & score_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ SEQ_{cr} & lb_{cr} & ub_{cr} & bool_{cr} & score_{cr} \end{bmatrix} \tag{4}$$

得分矩阵每一行表示一个大区间, 矩阵序列 *SEQ* 作为标识自动生成, *lb* (Lower bound)、*ub* (Upper bound) 分别表示区间的下界与上界, 布尔类型数据根据该区间得分高于或低于标准而确定, *score* 表示该区间的分数.

至此, 通过第一步划分, 得到了 *cr* 个新的区间, 记为 *RangeSecond*. 其中, 每个区间代表一个波峰或波谷, 并且有一个得分, 得分越高表示该区间内关于距离的元素更加密集.

步骤 5. 分区细化. 自顶向下的思想是在步骤 4 的粗略划分基础上, 按相同的方法继续细化.

步骤 6. 按得分比例, 分配式 (2) 中的 k.

通过上述工作, 获得集合 *Radius*, 并完善式 (2).

半径获取策略分为三部分, 首先确定在每个区间内的距离数目, 其次确定距离出现频率的波峰波谷; 最后执行距离的细化. 通过算法 1, 能获取针对数据集的几组重要距离值, 并记录在式 (2) 中.

2.3 点集覆盖模型

覆盖模型的思路是指在 *Radius* 下的相邻有效半径构成的覆盖区域能够覆盖整个点集的情况下, 计算覆盖多维体数目的比值. 具体地说, 点集覆盖模型给出了点集与 *Radius* 相邻半径的描述关系.

2.3.1 覆盖多维体

定义 4. 覆盖多维体 (Covering multidimensional cube). 覆盖多维体是指能够覆盖在点集欧氏空间 $\mathbf{R}^d$ 的一系列多维体, 表示为 $CMC(r)$. 点向量为二维时, 覆盖多维体为开圆; 点向量为三维时, 覆盖多维体为开球.

如点集为二维向量, 目标点 x 为 (d_1, d_2), 则空间内该点覆盖多维体涵盖的范围是以半径 r 形成的一个圆, 范围内的点 (x', y') 符合 $\{(x', y')|(x'-d_1)^2+(y'-d_2)^2 \le r^2\}$; 点集向量为 d 维, 则空间内该点覆盖多维体涵盖的范围是以半径 r 形成的多维空间, 范围内的点 $(x_1', x_2', \cdots, x_d')$ 为 $\{(x_1', x_2', \cdots, x_d')|(x_1'-d_1)^2 + (x_2'-d_2)^2 + \cdots + (x_d'-d_d)^2 \le r^2\}$

讨论 1. 覆盖多维体的覆盖原则.

覆盖原则为分割覆盖 (Division cover), 这是由于点分布的无序性造成的.

若覆盖多维体占据的空间为 *Space*, 则在点集中可寻找若干类覆盖多维体: $space_1, space_2, \cdots, space_s$ 使:

$$\bigcup_{i=1}^{s} space_i \subset Space \tag{5}$$

其中任意覆盖多维体 *space* 可由若干半径较小的子

覆盖多维体 (SubSpace) 构成, 需要满足:

$$space_i = \bigcup_{j=1}^{s'} SubSpace_j \tag{6}$$

覆盖多维体与覆盖原则如图 7 所示.

在图 7 中, 通过不同种类覆盖多维体, 使点集中的大部分点 (除孤立点外) 均被覆盖. 图 7 中的覆盖结果属于多种覆盖方法的结果之一.

需要注意的是: 分割覆盖原则使原先半径为 r_1, r_2 的覆盖圆分割为若干小的覆盖圆, 图 7 中被覆盖多维体划分后的点集被多种类型的覆盖圆 C 覆盖. 若规定 $\begin{cases} CMC_1 = C_1 = 2C_2 \\ CMC_2 = C_3 + C_4 \end{cases}$, 则该点集由确定的两类覆盖多维体 CMC_1, CMC_2 覆盖.

通过引入覆盖多维体的相关概念, 针对簇内孤立点的判定问题提出如下的解决思路.

目标问题: 在密度矩阵中, 确定并选择合适的一至多种密度, 或确定多种密度间的关联, 使推导出的参数在输入至 DBSCAN 算法后能够有效聚类, 并甄别孤立点.

转化结果: 如何选择半径, 使其构成的覆盖多维体在覆盖点集的条件下密度达到最高.

思路正确性探究: 将点集 P 视为含有多种密度混合的数据集 $\rho = \bigcup_{i=1}^{\delta} \rho_i$; 根据密度矩阵性质, 不同密度的覆盖多维体数量具有递减性, 即 $n_{\rho_1} > n_{\rho_2} > \cdots > n_{\rho_\delta}$; 并且每一个密度具有相对应的半径 r, 二元组 (ρ_δ, r_n) 展示了该半径下密度分布的主要区间. 因此, 点集存在等式:

$$\begin{aligned} m = & \underbrace{n_{r_1} \cdot (S_{r_1} \cdot \overline{\rho}_{r_1}) + n_{r_2} \cdot (S_{r_2} \cdot \overline{\rho}_{r_2}) +}_{\text{factor 1}} \\ & \underbrace{\cdots + n_{r_\delta} \cdot (S_{r_\delta} \cdot \overline{\rho}_{r_\delta})}_{\text{factor 2}} \end{aligned} \tag{7}$$

该等式由各类半径构成的覆盖多维体组成. 其中, m 为点的数目, $(S_r \cdot \rho_r)$ 为该密度覆盖多维体内点的数目, n_r 为对应覆盖多维体数量. 因式项 1 为点集中大部分正常点所占空间, 因式项 2 为松散或孤立点所占空间. 考虑因式项 1 作为主要研究目标, 使该组合达到数量 n_r 最小, 密度 ρ_r 最大. 为选择合适密度, 或确定密度间关联, 需要分析不同覆盖多维体数目的关系.

讨论 2. 点集空间能否被一类以上的覆盖多维体覆盖.

引理 1. 设点集 P 在 $\mathbf{R}^d$ 中是有界闭集, $\mathbf{C}$ 是覆盖多维体集合, 且 $\mathbf{C}$ 覆盖了点集 P. 则在点集 P 中必存在有限个覆盖多维体 $CMC_1, CMC_2, \cdots, CMC_s$, 同样完全覆盖点集 P.

解释: 1) 条件中, $\mathbf{C}$ 覆盖点集 P, 指 $\forall x \in P$, 存在覆盖多维体 $CMC \in \mathbf{C}$, 使得 $x \in CMC\,(r)$. 2) $\mathbf{C}$ 是指欧氏空间中的一系列邻域, 如 $\left(\frac{n}{n+1}, \frac{n+1}{n+2}\right)$, $n \in \{0, 1, 2, \cdots, N\}$, 实际就是覆盖多维体下的不同半径集合 $Radius$.

引理的证明与有限覆盖定理 (Heine-Boreltheorem) 证明相似, 本文不再证明. 最终可得到不同半径集合 $Radius$ 可覆盖点集 P 所在空间的结论. 因此需要讨论半径集合 $Radius$ 中不同半径覆盖点集的能力.

讨论 3. 覆盖点集空间时, 半径应满足的条件.

在半径 r 较小的条件下, 覆盖多维体能够完全覆盖点集, 有 $n \cdot \pi r^2 = n_p$. 若存在 $r' = r/\sigma$, 则存在 $n \cdot \pi\,(r')^2 = (1/\sigma^2)\, n \cdot \pi r^2 = n_p$, 由此 r' 同样能够实现点集的空间覆盖, 这不仅表明覆盖多维体的覆盖结果是多样的, 同样证明了当半径 r 小于某一条件时, 点集会被覆盖多维体完全覆盖.

设点集 P 在所处欧氏空间的任意维度 $dimension_d$ 的最大最小值分别为 $MaxV_d$, $MinV_d$,

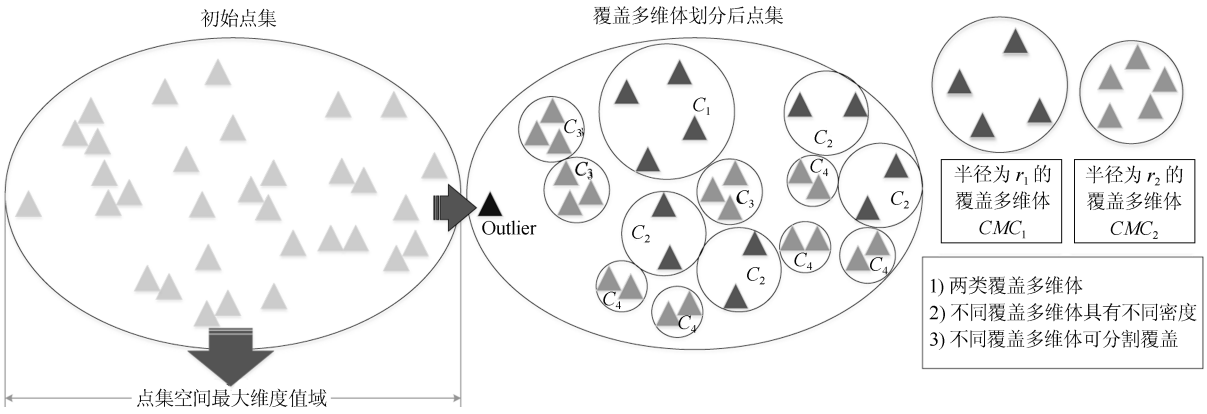


图 7　覆盖多维体分割覆盖原则示意

Fig. 7　The example of division cover in covering multidimensional cube

则认为 P 在该维度下的值域为 $[MaxV_d, MinV_d]$. 因此, d 维空间的每一维度下均有范围为 $[MaxV_d, MinV_d]$ 的多维欧氏空间.

当存在多维体, 半径 $r \in \mathbf{R}$, 且 $0 < r < 1/2Max(MaxV_d - MinV_d)$, $d = 1, 2, \cdots, d$. 即 r 应小于维度最大值域的一半. 为满足若干多维体覆盖点集, 有以下情况:

1) 当 $r = 1/2Max\left(MaxV_d - MinV_d\right)$, 则 $Space_{cmc} \cap Space_p = Space_p$, 表示该多维体能覆盖点集空间.

2) 当 $1/4Max\left(MaxV_d - MinV_d\right) \leq r < 1/2Max\left(MaxV_d - MinV_d\right)$, 则 $2 \cdot Space_{cmc} \cap Space_p = Space_p$, 表示两个多维体能覆盖点集空间.

3) 当 $r < 1/4Max\left(MaxV_d - MinV_d\right)$, 或 $r \ll 1/4Max\left(MaxV_d - MinV_d\right)$, 则 $n \cdot Space_{cmc} \cap Space_p = Space_p$, 表示 n 个多维体能覆盖点集空间.

其中 $Space_{cmc}$ 与 $Space_p$ 分别表示覆盖多维体与点集所占据的空间. 且当半径 r 小于最大维度值域 1/4 时, 覆盖的密度更高, 点集更加逼近 P 集合总数; 为此, 需要将计算的距离矩阵进行筛选, 留取较小的点间距离, 作为半径获取的学习数据.

这里将 r 的选择称为距离筛选 (Distance filter), 是 ODIC-DBSCAN 的第一项输入参数, 用来控制覆盖多维体的半径大小. 通过讨论 2, 讨论 3 可知, 任意点集 P 可由一类以上的覆盖多维体覆盖, 且为使覆盖密度更高, 半径也应具备一定的条件. 但点集被覆盖的方式具有多样性, 第 2.3.2 节将在满足点集被覆盖的条件下, 对不同类型覆盖多维体的数目关系展开讨论.

2.3.2 点比阈值

若仅仅表述第 k 密度与第 $k+1$ 密度的关系, 可以描述为以下一系列无差别曲线族, 定义为 $U(n_{\rho_{k+1}}, n_\rho) = E$, $k \geq 0$, 其中 E 为常数. 该曲线族描述了密度矩阵 $\boldsymbol{\rho}_{k\times m}$ 相邻密度间的关系, 如图 8 所示.

若设两类覆盖多维体的密度与对应半径分别为 (ρ_1, ρ_2) 与 (r_1, r_2), 则当两类覆盖多维体能够共同覆盖点集时, 所满足的数量关系如图 8 所示. 在图 8 中, 设曲线上任意点 p 为二元组 $(x\langle\rho_1, r_1\rangle, y\langle\rho_2, r_2\rangle)$ 构成, 则图中 p_1 与 p_2 在覆盖点集效果中是等价的.

对 U 上的任意一点 p, 满足:

$$m = \underbrace{n_{\rho_k} \cdot \overline{\rho}_k \cdot \pi r_k^2}_{\text{factor 1}} + \underbrace{n_{\rho_{k+1}} \cdot \overline{\rho}_{k+1} \cdot \pi r_{k+1}^2}_{\text{factor 2}} \tag{8}$$

$\overline{\rho}_k$ 表示每一行密度矩阵的平均密度; 其中等式左边第一个因式项表示以 $\overline{\rho}_k$ 为密度, 以 r_k 为半径的圆总共覆盖的点数目; 因式项 2 同理. 两个因式项之和等于点数目 m. 在这个约束条件下, 需要掌握不同覆盖多维体 $n_{\rho_k}/n_{\rho_{k+1}}$ 的关系.

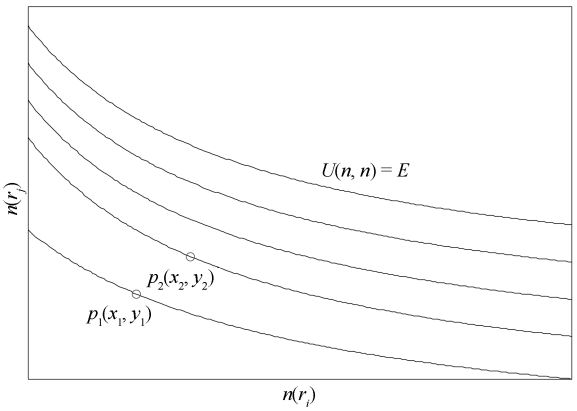


图 8 无差别曲线族

Fig. 8 Indiscriminate curve

定义函数 U 以描述密度 n_{ρ_k} 与 $n_{\rho_{k+1}}$ 间的均衡状态, 该均衡状态表示相邻半径间的偏向关系. 函数 U 应具备以下两点性质: 1) 函数为单调减函数; 2) 函数图像为下凹形状. 这两点性质符合无差别曲线的特性, 并以反比关系的形式展现.

$$U\left(n_{\rho_k}, n_{\rho_{k+1}}\right) = \left(\frac{\mu}{n_{\rho_k}} + \frac{\nu}{n_{\rho_{k+1}}}\right)^{-1}, \quad \mu > 0, \nu > 0 \tag{9}$$

函数 U 是 $y = 1/x$, $(x > 0)$ 的形式, 形态即以原点为对称, 反比函数双曲线的上部分. 从函数的构成上, 可以发现 n_{ρ_k} 与 $n_{\rho_{k+1}}$ 的系数为一对参数 (μ, ν), 该参数考虑了点集密度对相邻半径 r_i, r_j 的偏向性. 如在图 8 中, μ 较大时, 曲线偏向于 $n\left(r_i\right)$; ν 较大时, 曲线下凸则偏向于 $n\left(r_j\right)$. 参数通过寻找排序后密度矩阵的相邻密度差值最大的点, 并返回其排序序号获取, 表示为式 (10):

$$\begin{cases} \mu = \left\{i | i \in \max\left(\text{abs}\left(\rho_{\langle x_i, r_j\rangle}, \rho_{\langle x_{i+1}, r_j\rangle}\right)\right)\right\} \\ \nu = \left\{i | i \in \max\left(\text{abs}\left(\rho_{\langle x_i, r_{j+1}\rangle}, \rho_{\langle x_{i+1}, r_{j+1}\rangle}\right)\right)\right\} \end{cases} \tag{10}$$

为求多元函数 (9) 在约束条件 (8) 下的极值, 引入拉格朗日乘子法, 将 2 个变量与 1 个约束条件的最优化问题转为 $2+1$ 个变量的无约束优化问题. 在本例约束优化工作中, 构造拉格朗日函数:

$$L\left(n_{\rho_{k+1}}, n_{\rho_k}, \lambda\right) = U\left(n_{\rho_{k+1}}, n_{\rho_k}\right) + \lambda\left(\Psi\left(n_{\rho_{k+1}}, n_{\rho_k}\right)\right) \tag{11}$$

其中, L 为构造的拉格朗日函数, U 为待求优化问题, λ 为拉格朗日乘子, $\Psi\left(n_{\rho_{k+1}}, n_{\rho_k}\right)$ 为式 (8). 通

过引入函数 U, Ψ 拉格朗日函数为:

$$L\left(n_{\rho_{k+1}}, n_{\rho_k}, \lambda\right) = \frac{n_{\rho_k} n_{\rho_{k+1}}}{\mu n_{\rho_{k+1}} + \nu n_{\rho_k}} + \lambda\left(n_{\rho_k} \cdot \overline{\rho}_k \pi r^2 + n_{\rho_{k+1}} \cdot \overline{\rho}_{k+1} \pi r_{k+1}^2 - n_{point}\right) \tag{12}$$

则 $\frac{\partial L}{\partial \lambda}$ 为约束条件, 且 $\frac{\partial L}{\partial n_{\rho_k}}$ 与 $\frac{\partial L}{\partial n_{\rho_{k+1}}}$ 分别为:

$$\begin{cases} \dfrac{\partial L}{\partial n_{\rho_k}} = \dfrac{\mu n_{\rho_{k+1}}^2 + \lambda \overline{\rho}_k \pi r_k^2 \cdot \left(\mu n_{\rho_{k+1}} + \nu n_{\rho_k}\right)}{\left(\mu n_{\rho_{k+1}} + \nu n_{\rho_k}\right)^2} \\ \dfrac{\partial L}{\partial n_{\rho_{k+1}}} = \dfrac{\nu n_{\rho_k}^2 + \lambda \overline{\rho}_{k+1} \pi r_{k+1}^2 \cdot \left(\mu n_{\rho_{k+1}} + \nu n_{\rho_k}\right)}{\left(\mu n_{\rho_{k+1}} + \nu n_{\rho_k}\right)^2} \end{cases} \tag{13}$$

令 $\frac{\partial L}{\partial n_{\rho_k}}$ 与 $\frac{\partial L}{\partial n_{\rho_{k+1}}}$ 为 0, 则有联立式 (14):

$$\begin{cases} \mu n_{\rho_{k+1}}^2 = -\lambda \overline{\rho_k} \pi r_k^2 \cdot \left(\mu n_{\rho_{k+1}} + \nu n_{\rho_k}\right) & \text{(14a)} \\ \nu n_{\rho_k}^2 = -\lambda \overline{\rho}_{k+1} \pi r_{k+1}^2 \cdot \left(\mu n_{\rho_{k+1}} + \nu n_{\rho_k}\right) & \text{(14b)} \end{cases}$$

将 (14a) 与 (14b) 作比, 则有结果:

$$\frac{n_{\rho_{k+1}}}{n_{\rho_k}} = \sqrt{\frac{\nu \overline{\rho}_k r_k^2}{\mu \overline{\rho}_{k+1} r_{k+1}^2}} \tag{15}$$

对式 (15) 的研究从以下几点分析.

1) 考虑式 (15) 的构造. 通过式 (15) 可以发现, $n_{\rho_{k+1}}/n_{\rho_k}$ 的值由参数 (μ, ν), 平均密度 $\overline{\rho}_k$, $\overline{\rho}_{k+1}$, 半径 r_k, r_{k+1} 因子确定. 首先在这些因子中, 半径的影响程度更大一些. 其次, $n_{\rho_{k+1}}$ 所占比例大小由第 k 密度中的 $\bar{\rho}_k r_k^2$ 决定, 而非第 $k+1$ 密度决定; n_{ρ_k} 则相反. 这体现了数目之比受到相邻密度间的相互约束, 其原因是由于 $n_{\rho_{k+1}}$ 所代表的较大半径圆的数目实际应小于 n_{ρ_k} 所代表的较小半径圆的数目, 仅在密度相差极大的情形下 $\left(\overline{\rho}_k \gg \overline{\rho}_{k+1}\right)$, 即当密度的影响程度与半径平方的影响程度逼近时, $n_{\rho_{k+1}}$ 才会靠近 n_{ρ_k}, 但在此时通常也会考虑在 $k+1$ 密度下是否存在孤立点. 通过分析式 (15) 的结果, 通常有 $\mu > \nu, \overline{\rho}_{k+1} < \overline{\rho}_k, r_{k+1} > r_k$ 在此基础上, $n_{\rho_{k+1}}/n_{\rho_k}$ 通常小于 1.

2) 考虑式 (15) 的含义. a) 本身含义. $n_{\rho_{k+1}}/n_{\rho_k}$ 表达式的含义是在点集 P 中, 以第 $k+1$ 密度代表的半径 r_{k+1} 构成的覆盖多维体 $CMC\left(r_{k+1}\right)$ 的数目与第 k 密度代表的半径 r_k 构成的覆盖多维体 $CMC\left(r_k\right)$ 数目之比. 实际上这表示了点集 P 中点分布关系由 $n_{\rho_{k+1}}$ 个密度为 $\overline{\rho}_{k+1}$ 的覆盖多维体与 n_{ρ_k} 个密度为 $\overline{\rho}_k$ 的覆盖多维体之和构成, 即两类覆盖多维体的面积总和与点集空间相等或逼近. ②引申含义. $n_{\rho_{k+1}}/n_{\rho_k}$ 表示了全局 P 中较松散点 $\overline{\rho}_{k+1}$ 的组数与较紧密点 $\overline{\rho}_k$ 的组数的比值, 可视为密度为 $\overline{\rho}_{k+1}$ 密度类型的点数目与密度为 $\overline{\rho}_k$ 密度类型的点数目的比值. 因此, 将该比值视作以某个点为对象, 半径 r_k 与半径 r_{k+1} 下点数目比值的比较对象, 并将该阈值称为点比阈值 (Point ratio threshold, PRS).

$$\begin{cases} \dfrac{n_{\rho_{k+1}}}{n_{\rho_k}} \le pt_{x_i} & \text{(16a)} \\ \dfrac{n_{\rho_{k+1}}}{n_{\rho_k}} > pt_{x_i} & \text{(16b)} \end{cases}$$

式中, $n_{\rho_{k+1}}/n_{\rho_k}$ 表示在相邻密度下的点比阈值, pt_{x_i} 表示以点 x_i 为研究对象, 计算出该点在相关半径下的密度比. 在此条件下, 式 (16a) 表示 x_i 的分布的外围点密集程度大于平均点比阈值, 说明该点高密度, 且分布于簇的中心部分; 式 (16b) 则表示该点在相关半径下密度差异较大, 该点处于簇的边缘区域.

当点集存在 k 层密度曲线, 即 k 个无差曲线族时, 则相应会出现 $k-1$ 个相邻的点比阈值, 称为点比阈值组 (Group of point ratio thresholds, GPRT).

2.4 ODIC-DBSCAN

ODIC-DBSCAN 算法的核心思想是将点比阈值组作为算法的考察阈值, 确认核心对象在规定的两个半径内密度变化是否大于阈值, 如果密度变化大于规定阈值, 则说明该点外围的点较多, 该点并非处于簇边缘; 反之则证明该点处于簇边缘部分, 因此不必将其纳入扩充区域, 也不必将其标记为该簇. 因此重定义的核心对象如下:

定义 5. 核心对象 (Core object). 若以 x_i 为研究对象, 在集合 $Radius$ 中相邻半径 r_α, r_β 下点的数目之比大于预处理的点比阈值, 则称 x_i 为核心对象.

在任意非均匀分布点集下, 无论是否存在模糊边界, 当限制条件 (如原 DBSCAN 中的 $epsilon$ 与 $MinPts$, 或本文提出的点比阈值) 逐渐步进时, 簇间会渐渐产生靠拢、聚合的现象, 直至最终汇聚为一个簇. 该现象被称为簇的融合. 簇的融合将模糊边界融聚在一起, 因此难免会对簇的聚类结果产生误判. 因此孤立点判定方法需要不断循环、步进点比阈值, 直到簇发生了融合现象, 便停止迭代. 此时对没有簇号的点, 可记为孤立点. 在检测方法中, 包含两个步骤:

步骤 1. 通过算法 3 确立点集存在的簇群.

在簇群的建立工作中, 给定一个有效点阈值 ($Effective\ \ Points\ \ Threshold$), 表示没有簇编号的点 (孤立点) 占点集数目的比值, 若高于有效点阈值, 则确定簇群建立成功, 否则失败 (可参照附录算法 3).

有效点阈值 $Effective\ Points\ Threshold$ 是 ODIC-DBSCAN 算法的第二个参数. 该阈值的取值完全是有限的, 可每隔十个百分点取一数值. 实际上, 阈值的大小规律是有迹可循的, 簇的紧密程度与该值有明显的关联: 当簇的分界清晰, ($Effective\ Points\ Threshold$) 取 90 % 依然无误; 当簇间模糊, 点分布无明显规律, 则将该参数调至 50 % 以下即可.

步骤 2. 迭代执行 ODIC-DBSCAN 算法, 每次增加不同半径下点数目的比例, 直至发现簇产生融合现象时停止.

孤立点的检测关心首次确立的簇群是否发生了融合现象, 如果没有发生融合, 则记录相关数据, 循环算法; 否则返回上一次的 ***IDX***, 作为聚类的最终结果. 算法通过获取并匹配 ***IDX*** 的分布, 确认簇是否发生融合, 并返回结果. 其流程如图 9 所示.

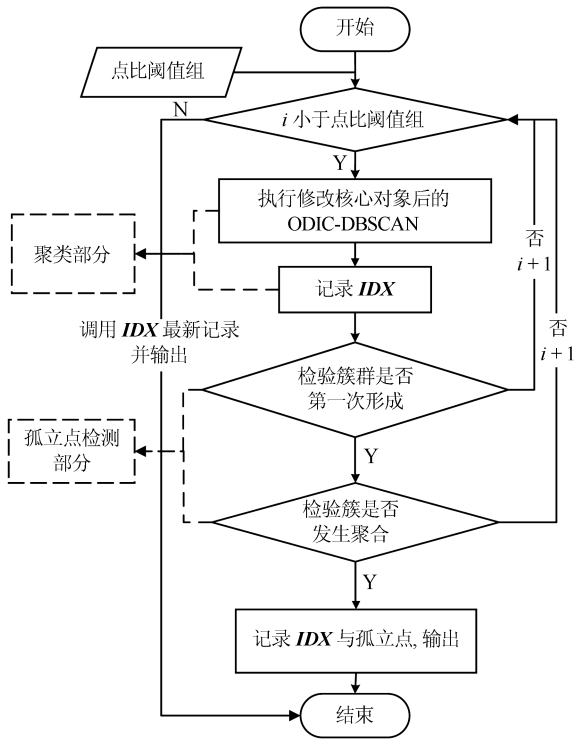


图 9 孤立点检测流程

Fig. 9 The procession of outlier detection

2.5 时间复杂度分析

程序结构被分为以下几个部分: 1) 距离矩阵; 2) 筛选符合要求的距离; 3) 半径获取策略; 4) 密度矩阵; 5) 计算点比阈值; 6) ODIC-DBSCAN 算法.

1) 距离矩阵时间复杂度为 $\mathrm{O}(n^2)$.

2) 筛选符合要求距离的时间复杂度为 $\mathrm{O}(n)$; 遍历距离一次, 找出符合要求的距离点并组成数组.

3) 半径获取策略的时间复杂度为 $\mathrm{O}(n)$; 且此时 n 已远小于 "筛选符合要求距离" 中的 n, 其时间主要消耗在选定区域内的距离数量中.

4) 密度矩阵时间复杂度为 $k \cdot \mathrm{O}(n)$. 其中 n 为点集规模, k 为集合 $Radius$ 中有效半径的数目.

5) 计算点比阈值, 线性时间复杂度.

6) ODIC-DBSCAN 算法时间复杂度为 $k \cdot \mathrm{O}(n)$. 实际上, 距离矩阵作为参数直接传入 DBSCAN, 无需重复计算. $\mathrm{O}(n)$ 为对 n 个点进行遍历, 寻找密度相连的点; k 为外层循环, 循环体为点比阈值组.

通过以上分析, ODIC-DBSCAN 算法时间复杂度的数量级与 DBSCAN 相同, 均为 $\mathrm{O}(n^2)$.

3 实验与分析

本文实验分为三个部分: 第一部分为半径获取策略的效果展示; 第二部分为 ODIC-DBSCAN 的性能测试: 包括在 UCI 真实的高维数据中检测参数敏感性与时间效率; 第三部分为簇内与簇间孤立点检验效果对比.

实验将 DBSCAN、AP、DPC、ReCon-DBSCAN、ReCon-OPTICS 作为簇内孤立点检测的对比算法; 此外, 将 LDOF、F-ABOD、LOF 以及 OPTICS 四种算法作为簇间孤立点检测性能的对比算法, 并使用公开数据集与模拟数据集. 在性能测试与簇间孤立点检测中, 选择 UCI 公开的 3 个真实数据集与聚类公开的 3 类 benchmark; 在簇内孤立点检测中, 制定了相应的模拟数据集 1 ∼ 3, 其数据分布将在对应部分给予说明.

3.1 半径获取策略

在验证半径获取策略的方法上, 两个具有代表性的数据集. 图 10 (a) 含有两个簇, 且距离差为 2. 图 10 (b) 含有三个簇, 其左下方两个簇与图 10 (a) 相同, 最远两个簇之间距离差为 10. 图 10 中每个簇包含 100 个点, 且均以随机数的方式生成.

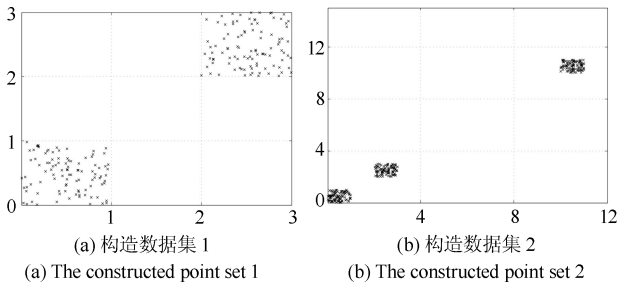


图 10 构造数据集散点图

Fig. 10 The constructed point sets scatters

通过图 11, 可以观察距离的数目与其值的关系, 该图是由算法产生 $1 \times 2\,000$ 的矩阵 ***DistRatio*** 构

成的. 图 11 中在 1～2 的距离内, 有一个明显的直线上升. 其中, 在上升段的左侧, 是两个簇内各自点之间的两两距离; 在上升段的右侧, 是两个簇之间的相互距离. 左右之间的上升跃进是半径获取策略的主要作用, 屏蔽了一些 “没有作用” 的距离, 这些距离会使算法产生大量的冗余, 因此半径获取策略将这些距离筛选、排除. 相应的统计下, 在 0～1 范围内, 划分出的距离数目为 950 个; 在 1～2 范围下, 算法划分的距离数目 86 个; 而 2～3 下, 数目为 661 个, 3～4.5 中, 距离数目为 430 个.

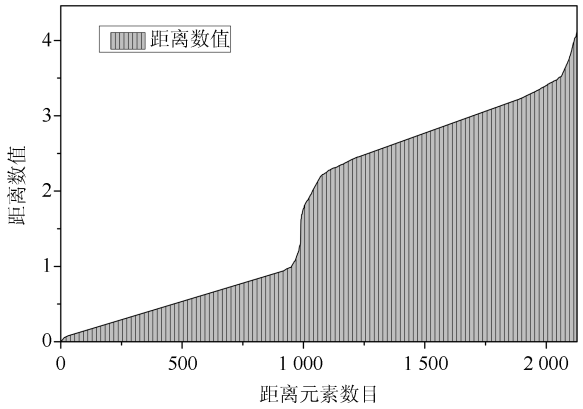


图 11　数据集 1 下的距离分割

Fig. 11　The distance division of point set 1

图 12 为分割图 10 (b), 算法划分的距离产生的结果. 图中的划分出现了多层梯度, 按从低至高的顺序, 梯度的解释如表 1 所示.

表 1　梯度含义

Table 1　The meaning of gradient

位置	意义
第一梯度左部	以三个簇各自内部点的距离为对象
第一梯度右部	以簇 1、簇 2 间的距离为对象
第二梯度右部	以簇 2、簇 3 间的距离为对象
第三梯度右部	以簇 1、簇 3 间的距离为对象

通过图 11、12 的数据, 可验证半径获取策略能够选择适应于数据集的几组重要距离, 输出的结果也具有针对性.

3.2　ODIC-DBSCAN 性能测试

本节将对 ODIC-DBSCAN 的性能做一些典型测试, 包括 1) 参数敏感性分析; 2) 运行时间分析. 数据集的采用说明如下:

1) 实验选择 UCI 公开的数据集 (http://archive. ics.uci.edu/ml/datasets.html): Banknote Authentication (BA)、Chess 与 Breast Cancer Wisconsin (WDBC). 在三类数据集的预处理中采取同样的方法: 删除反类中 (孤立点类) 绝大部分数据, 仅保留 10 条反类样本作为孤立点进行检测. 三个数据集的样本数目与维度分别为: 772 × 4、1 679 × 36、367 × 30.

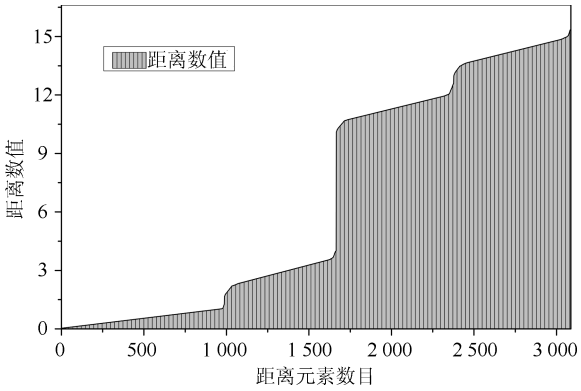


图 12　数据集 2 下的距离分割

Fig. 12　The distance division of point set 2

2) 在时间分析中, 选择聚类的三种 benchmark (https://cs.joensuu.fi/sipu/datasets), 分别测试不同规模, 不同簇数目, 以及不同维度下的运行时间.

3.2.1　参数敏感性分析

ODIC-DBSCAN 参数为: 1) 距离的筛选条件; 2) 簇形成时的簇阈值. 通过 3 类数据集, 在控制两类参数的情况下, 分析数据集产生的孤立点数目.

表 2 展示了在真实的高维数据集下, 调整不同参数下的孤立点检测结果. 从表中数据可知, ODIC-DBSCAN 在不同参数的影响下, 对数据分析较为稳定, 检测结果对参数并不敏感.

分析可知: 1) 距离筛选参数将距离细化后再选择数据集中重要半径, 但对不同数据集有不同的要求. 如数据集 2, 当半径为最大距离的 1/5 时, 由于半径过小, 则不能产生聚类效果. 2) 有效点阈值参数对结果的影响是具有规律的, 该参数表示高于有效点占全局点的比例时, 簇的产生生效. 因此可以发现, 随着该参数的增加, 孤立点在不断减少, 而到达一定程度时, 结果将保持不变; 在该情况下, 产生的孤立点数目是稳定且可靠的.

另外, 当有效点阈值降低时, 孤立点数目增长. 实际上, 有效点阈值降低时, 簇发生了融合, 而较多被检测的孤立点来源于簇之间的边缘点对象. 这说明 ODIC-DBSCAN 对图 1 中的现象 1 和 2 具有检测效果.

3.2.2　算法运行时间分析

算法的时间性能关系到应用效率. 本节选择了: 1) 三类 benchmark, 包括不同规模, 簇数目以及维度的数据对象; 2) 两种密度聚类算法 (DPC 与 AP 算法), 使之作为对比方法, 并控制迭代次数.

1) 不同规模

图 13 展示了不同算法在不同规模数据集下的运行时间. DPC 算法的运行默认自动选取所有的簇心, 并无需迭代, 算法执行仅需要计算每个点的局部密度, 并寻找该点范围内更高的局部密度对象即可. 而 AP 算法虽然执行时间有限, 但为获得聚类中心的收敛, 需要不断迭代. ODIC-DBSCAN 虽然需要进行外层循环迭代, 不断遍历全局点, 并根据核心对象条件执行扩充方法; 但是该方法在预处理中选择了针对数据集的重要半径, 因此时间低于 AP 算法.

2) 不同簇数目与不同维度

在不同簇数、维度的数据下, 各类算法的运行时间如图 14、15, 且三类 benchmark 在数据规模上相同.

从数据中可以观察到, 最为重要的变化是 ODIC-DBSCAN 的速度在簇数目与维度的 benchmark 中得到了极大的提升, 这主要是因为数据的分布对算法具有较大的影响. 因此在相同规模下, 数据的分布对运行时间的影响不可忽略. 在图 13 的数据中, 分布总是包含 95 个簇, 形状复杂; 因此 ODIC-DBSCAN 在运行时需要考虑任意两个簇之间的距离问题, 运行时间较长; 而在图 14 中, 每个 benchmark 对应的簇数目较少, 分布较简单, 运行时间也较短.

为了更清晰地观察算法在运行时间的细节, 启用了 MATLAB 时间运行探查器. 在此对比图 13 中数据量为 5 000 的数据集, 以及图 14 中簇数目为 5 的数据集; 两个数据集的点数目完全相同, 但数据分布不同; 另外, 控制迭代的次数为 1 次.

从表 3 中可发现, 在同样规模的数据集下, 由于数据分布不同, 算法运行的时间也受到了影响. 通过表中数据, 可知时间主要消耗在 “合并邻居” 这一条命令行中 (算法 2 第 20 行). 该行代码负责扩展簇, 即确定一个核心对象后, 寻找其邻居, 然后将寻找邻居返回的结果继续作为输入, 并继续寻找邻居, 直到结束条件. 可以发现, 簇的数目较多时, 合并的次数也较多, 但算法时间消耗也随之上升, 这是因为原邻居数组的长度与新纳入的点数组长度都有所增加, 合并数组的时间消耗也大幅度增长.

表 2 距离筛选与有效点阈值的敏感性测试

Table 2 The sensitivity of distance filter and effective points threshold

距离筛选	BA		Chess		WDBC	
	有效点阈值	孤立点数目	有效点阈值	孤立点数目	有效点阈值	孤立点数目
2	0.2	552	0.2 ∼ 0.6	645	0.2 ∼ 0.9	6
	0.3 ∼ 0.9	8	0.7 ∼ 0.9	136		
3	0.2	545	0.2 ∼ 0.6	645	0.2 ∼ 0.9	6
	0.3 ∼ 0.9	27	0.7 ∼ 0.9	136		
4	0.2 ∼ 0.3	497	0.2 ∼ 0.6	645	0.2 ∼ 0.9	6
	0.4 ∼ 0.8	86	0.7 ∼ 0.9	136		
	0.9	21				
5	0.2 ∼ 0.3	497	–	–	0.2 ∼ 0.9	6
	0.4 ∼ 0.8	86				
	0.9	9				
6	0.2 ∼ 0.3	497	–	–	0.2 ∼ 0.9	6
	0.4 ∼ 0.8	143				
	0.9	9				

表 3 ODIC-DBSCAN 在相同规模不同分布的数据集下时间运行细节

Table 3 Time details of ODIC-DBSCAN on the point sets that have same scale but different distributions

数据集	总时间 (秒)	预处理 (秒)	ODIC-DBSCAN (秒)			
			ExpandCluster (运行次数/时间/所占该函数百分比)			其他
			查询邻居	计算比值	合并邻居	建立簇等
95 个簇	14.719	4.339	4 999/0.399/4.3 %	4 999/0.715/7.6 %	4 999/6.511/69.6 %	0.04
5 个簇	3.718	1.631	2 980/0.182/28.4 %	2 980/0.378/59.1 %	2 980/0.010/1.6 %	0.43

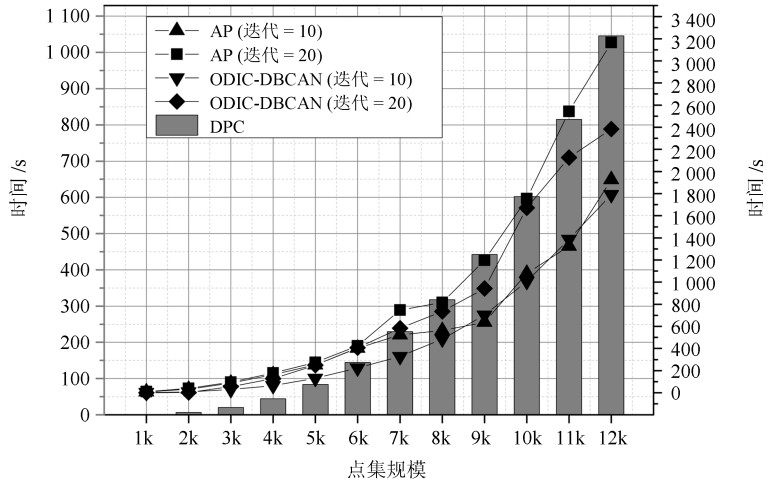


图 13　不同算法在不同规模 benchmark 下的运行时间

Fig. 13　Time of algorithms on scale benchmark

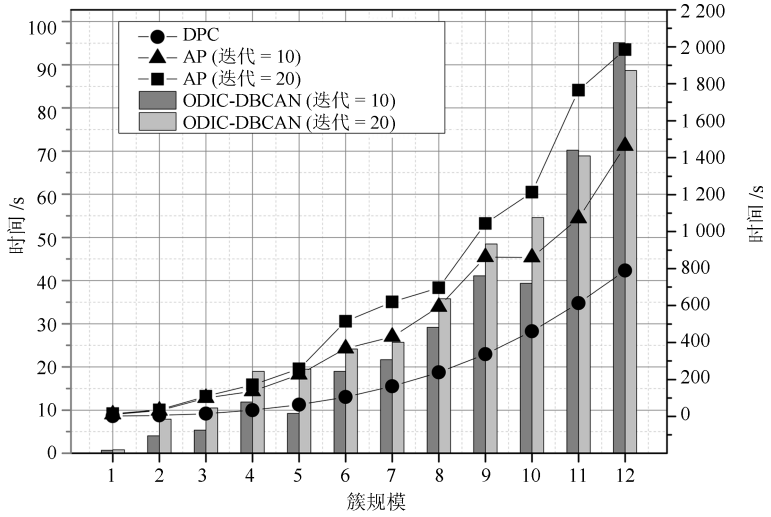


图 14　不同算法在不同簇数目 benchmark 下的运行时间

Fig. 14　Time of algorithms on cluster benchmark

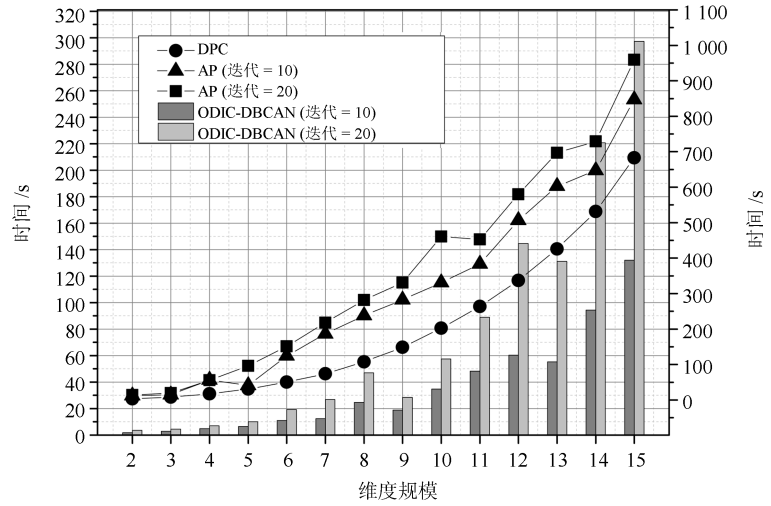


图 15　不同算法在不同维度 benchmark 下的运行时间

Fig. 15　Time of algorithms on dimension benchmark

3.3 孤立点的检测与对比

在本节中, 与基于密度聚类的方法进行对比, 使用 DBSCAN、AP、DPC、ReCon-DBSCAN、ReCon-OPTICS 五类算法在参数不同的情况下分别检验其是否能检测出预设的孤立点, 并与 ODIC-DBSCAN 进行比较.

1) HR (Hit rate). 该值表示算法的命中率. 设点集 P 中所有孤立点构成集合 $Outliers$, 二值算法检测出的孤立点为集合 DO_b (Detected outliers of binary results), 非二值检测出的孤立点为 DO_{nb} (Detected outliers of non-binary results), 则

$$HR = \begin{cases} DO_b \cap Outliers, & binary \quad type \\ DO_{nb} \cap Outliers, & non-binary \quad type \end{cases}$$

2) $SRoPO$ (Sum rank of pre-set outliers).

定义 6. 预设孤立点排序和 ($SRoPO$, Sum rank of pre-set outliers). 设点集 P 内包含 n_o 个孤立点, 非二值检测算法中任意点与其离群程度构成 $\langle x_i, outlier-ness_i \rangle$ 二元组, 将 P 内所有点按 $outlier-ness$ 由高至低排序, 则任意点均包含孤立程度的排列序号 $rank_i$; 此时有 $SRoPO = \sum_{i=1}^{n_o} rank_i$

表 4 展示了第 3.3.1 节与第 3.3.2 节特殊符号的意义.

表 4 特殊符号与其意义

Table 4 Symbols and its meanning

符号	意义
k	近邻参数数目
Top-n	查准率, 前 n 个检测结果中包含几个预设孤立点
per	DPC 算法的截断距离 $percentage$ 表示所有点的相互距离中由小到大排列占总数的百分比
pre	AP 算法参数, 表示数据偏好 $preference$, 用来确定簇数目
cn	簇数目 $cluster\ number$
$para$	ODIC-DBSCAN 参数 $parameter$: (距离筛选, 有效点阈值)
$ToDR$	ReCon-DBSCAN, ReCon-OPTICS 两类算法参数: Threshold on density ratio 表示密度比阈值
DO, nDO	检测出的孤立点集合与其对应数目
DI, nDI	检测出的簇内孤立点集合与其对应数目

3.3.1 簇内孤立点检测对比

数据集设计: 为了检验 ODIC-DBSCAN 在上述条件下的簇内孤立点检测效果, 重新制定了三个模拟数据集, 该数据集的形态如图 16 所示.

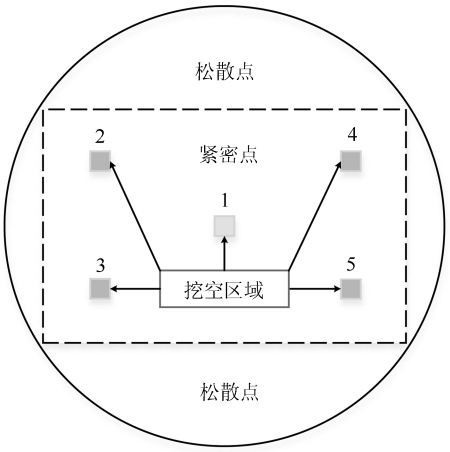


图 16 构造数据集

Fig. 16 Construction of point sets

在图 16 中, 外圆是点集的范围, 在圆内设置一内接矩形, 矩形内部安置了密集的数据, 矩形外部与外圆间安置了松散的数据. 数据集 1) 外圆处点 527 个, 矩形内部安置点 5 082 个, 将范围 1 下的点清除, 并添加点 (0.025, 0.025). 数据集 2) 外圆处点 546 个, 矩形内部点 14 385 个, 将 1～5 范围内的点清除, 添加点 $\{(0.025, 0.025), (-0.725, -0.225), (-0.775, 0.225), (0.775, -0.225), (0.775, 0.225)\}$, 数据集 2 共计点 15 266 个. 数据集 3) 外圆处点 557 个, 矩形内部 19 776 个点, 该点集的处理方式与点集 2 相同, 共计点 20 078 个.

1) ODIC-DBSCAN

图 17～19 展示了 ODBC-DBSCAN 在模拟数据集 1～3 中的结果. 观察图 17 结果可发现, 聚类将密集区的点聚合出来, 稀疏点输出为孤立点. 在数据密集区中, 由于存在点 5 552 个, 与挖空区域面积 (0.025) 相比, 数据集的点依然有相对稀疏的特性, 因此算法产出了与预设孤立点相近的点.

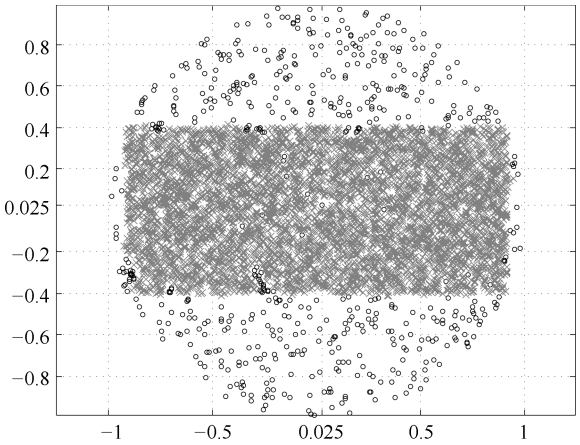


图 17 数据集 1 下的 ODIC-DBSCAN 算法结果

Fig. 17 Results of ODIC-DBSCAN in point set 1

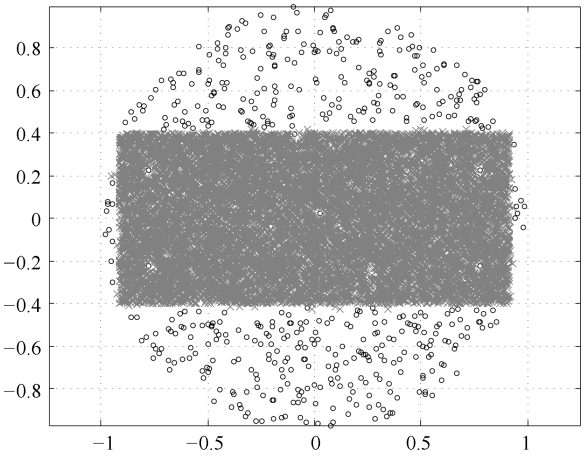


图 18　数据集 2 下的 ODIC-DBSCAN 算法结果

Fig. 18　Results of ODIC-DBSCAN in point set 2

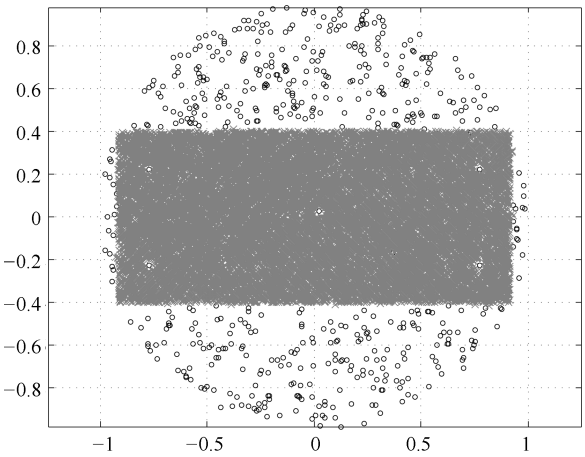


图 19　数据集 3 下的 ODIC-DBSCAN 算法结果

Fig. 19　Results of ODIC-DBSCAN in point set 3

这些孤立点的环境相仿, 与周围均有不同程度的空白, 因此聚类结果将他们分离出来. 图 18 中, ODIC-DBSCAN 算法不仅提供了正确的聚类结果, 同样分辨出六个孤立点. 这六个孤立点中有五个是预先设定的, 另外一个在簇的边缘处. 这说明随着簇密集程度的增加, 孤立点的分割也显得越来越清晰. 在图 19 中, 数据集 3 密集区的密度更大, 在此情况下, 设定的五个孤立点较为明显. 从聚类结果可以看出, ODIC-DBSCAN 在高密度下识别出了簇与五个孤立点.

2) DBSCAN

图 20 展示了在不同参数 (不同 $MinPts$) 下模拟数据集 1～3 的 DBSCAN 聚类结果. DBSCAN 在不断调参 (控制 $epsilon$ 不变, 调整 $MinPts$) 后, 三类数据集均产生了分裂的现象. 当数据密集区越为紧密, 裂变的效果越明显. 这是因为 DBSCAN 的核心注重簇的扩充, 而由于 $MinPts$ 条件限制, 导致密集区被分裂为多个簇. 而在这之间, 预设的孤立点始终不能被检测出, 这是因为算法仅能接收一种密度条件. 而 ODIC-DBSCAN 具有较强的检测能力, 是由于我们对 DBSCAN 算法的核心对象重新进行了定义, 优化了簇扩充的条件, 从点的数量, 转移到点周围密度的变化. 不仅如此, ODIC-DBSCAN 算法能够自学习整个点集中的密度分布, 提供合理的距离测试与密度变化. 通过以上支持, 使孤立点的检测结果较为理想.

3) DPC 与 AP

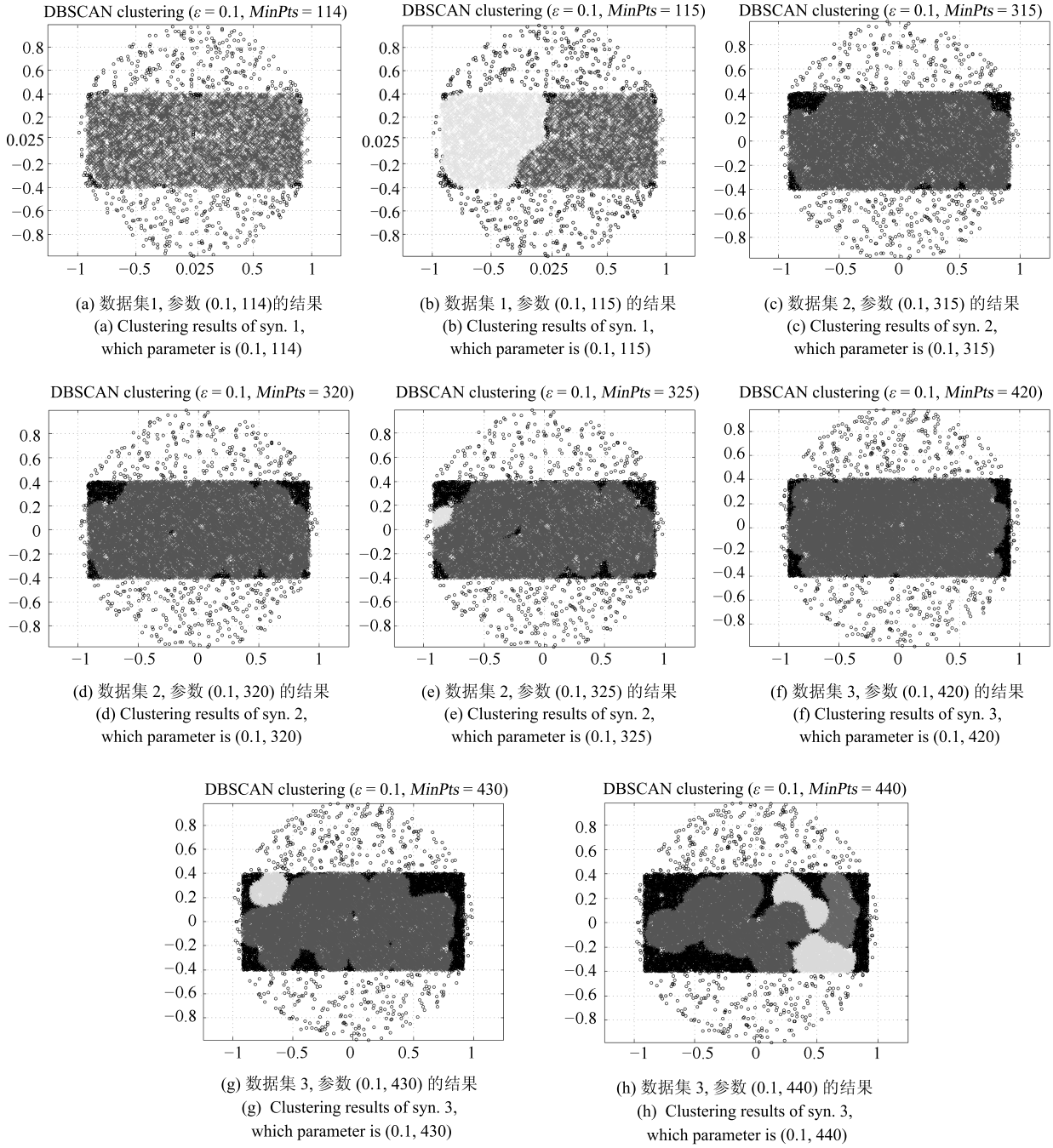
DPC 与 AP 聚类下对孤立点的识别结果如表 5 展示. DPC 算法中, 选择 ρ 较小, δ 较大的对象为孤立点. 在分析中, 选择排序后的 ρ/δ 最小的 n 个值作为分析对象. 其中 $position$ 表示预设的孤立点的 ρ/δ 值在正常点内的排序号码. 另外, CORE 表示属于每个簇的点数目. 其次, 有另外一些点在 CORE 集合的边缘处, 这些点被称为 HALO.

实验发现, 调节 per 在 1 %～2 % 区间内, 算法明显产生了两个簇心; 算法将高密区域分为不同的部分, 而非一个整体. 这说明 DPC 算法对参数是敏感的, 首先受限于参数 per 的调控, 不同的截断距离 d_c (Cutoff distance) 将数据的密度状态划分为不同的结果; 其次是簇心的选择, 完全改变了聚类结果. 在该实例中, 发现高密与低密区域的界限很难区分, 而仅用 HALO 表述, 说明算法的参数不能很好地适应不同密度下的数据集聚类结果. 也因而未能有效检测预设孤立点. 另外, DPC 算法能检查簇间孤立

表 5　DPC 与 AP 在模拟数据集 1～3 的检测结果

Table 5　Detection results of DPC and AP on synthetic point sets 1～3

Point sets	DPC					AP		
	cn	*per*	CORE	HALO	*position*	*pre*	*cn*	n_{DI}
Synthetic 1	2	1 %	1 585:1 411	1 278:1 805	78	−200	3	–
	2	2 %	1295:1 664	1 358:1 672	162	−300	2	–
Synthetic 2	2	1 %	1 009:1 105	7 197:5 955	92 82 68 75 65	−100	6	–
	2	2 %	1 668:1 228	6 425:5 945	89 62 60 110 64	−150	5	–
Synthetic 3	2	1 %	2 688:1 631	7 770:7 989	48 40 58 32 37	−100	8	–
	2	2 %	5 082:688	10 028:4 280	58 39 37 34 47	−200/− 300	4	–



(a) 数据集1, 参数 (0.1, 114)的结果
(a) Clustering results of syn. 1, which parameter is (0.1, 114)

(b) 数据集 1, 参数 (0.1, 115) 的结果
(b) Clustering results of syn. 1, which parameter is (0.1, 115)

(c) 数据集 2, 参数 (0.1, 315) 的结果
(c) Clustering results of syn. 2, which parameter is (0.1, 315)

(d) 数据集 2, 参数 (0.1, 320) 的结果
(d) Clustering results of syn. 2, which parameter is (0.1, 320)

(e) 数据集 2, 参数 (0.1, 325) 的结果
(e) Clustering results of syn. 2, which parameter is (0.1, 325)

(f) 数据集 3, 参数 (0.1, 420) 的结果
(f) Clustering results of syn. 3, which parameter is (0.1, 420)

(g) 数据集 3, 参数 (0.1, 430) 的结果
(g) Clustering results of syn. 3, which parameter is (0.1, 430)

(h) 数据集 3, 参数 (0.1, 440) 的结果
(h) Clustering results of syn. 3, which parameter is (0.1, 440)

图 20 不同参数下对模拟数据集 1～3 的 DBSCAN 聚类结果

Fig. 20 DBSCAN Clustering results of point sets 1～3 with different parameters

点, 但是对簇内孤立点难以检测; 这同样是因为 d_c 对检测具有大的干扰, 因为该值极大地影响了局部密度, 由于簇内不同的点密度差较小, 因此对该值的敏感很高. 而 AP 算法的本质是不断更新、计算相似度矩阵中每个点的吸引度信息与归属度信息, 并寻找簇中心的过程; 因此设置的模拟数据集在簇的划分上具有对称与均匀的特性; 由于算法在迭代中检查聚类中心的变化, 因此对簇内的孤立点并不敏感. 实际上, 在低密区域设置的孤立点也被纳入了正常簇中. AP 算法在点与点的信息传播中具有很强的特性, 但是在孤立点的判定中, 边缘点由于对簇内点具有归属度, 因此容易被纳入在簇的内部.

4) Re-Con DBSCAN 与 Re-Con OPTICS

表 6～7 展示了基于密度比方法中的 Re-Con DBSCAN 与 Re-Con OPTICS 算法在 Synthetic1-3 数据集中的检测结果. 该方法是由于 Reconditioning approach 提供了一种基于密度比的策略, 与本文提出的 ODIC-DBSCAN 具有相近之处. 表中参

数为该方法的输入, $\langle eps, eta\rangle$ 参数对表示构造密度比所需的不同大小半径.

从表 6 中的数据可知, 算法在不同参数的影响下效果不同, 随着阈值的增加, 检测出的孤立点数目也逐渐增加, 且对预设的簇内孤立点有一定的鉴别能力. 这是因为该方法采用的密度比思想能够针对不同密度抽取不同的密度阈值, 并能够阻隔多重密度带来的干扰; 但是, 该方法对参数具有依赖性, 体现在表中相同数据集下不同参数的结果. 另外, 由于预设的簇内孤立点空间空白区域较小, 且受到密度稀疏区域的干扰, 因此给定的密度阈值不能满足簇内孤立点密度的特殊条件.

在表 7 OPTICS 密度比算法中, $ToDR$ 表示算法构造的两层范围下近邻数目之比, 因此约定内层范围邻居数目 $MinPts$ 为 8, 则 $ToDR = 10$ 时, 外层范围邻居数目为 80. 对数据集 1～3, 构造的密度比阈值 $ToDR$ 越大, 命中率 HR 越大, 这是因为当构造的两层范围间密度差越大时, 低密区域的孤立点越容易被检测. 另外, 命中率涨幅由快变慢, 这是由数据的分布造成的, 模拟数据集的密度大致有两类, 因此涨幅在 $ToDR$ 为 10～30 间变化后, 孤立点检测能力的增加逐渐变缓.

对比表 6 与表 7 可发现, OPTICS 的密度比算法对簇内孤立点并不敏感, 若将密度比阈值增加, 预设的簇内孤立点与周围点的密度因为不满足较大的阈值而被算法忽略; 若将密度比阈值减小, 则模拟数据集的低密区域内部分点会被认作正常数据, 导致算法检测能力降低.

3.3.2　簇间孤立点检测对比

本节为比较 ODIC-DBSCAN 与其他几种密度聚类方法以及孤立点检测方法对簇间孤立点的检验能力, 使用 LOF、LDOF、F-ABOD 三类孤立点检测算法, 与 OPTICS、DPC、ReCon-DBSCAN、ReCon-OPTICS 四类密度聚类方法进行对比. 在数据集的采用上, 选择第 3.2 节所述三类公开高维数据集. 对比实验结果如表 8 所示.

表 8 展示了二值与非二值检测的六种方法在公开数据集 1～3 的检测结果. 在非二值检测的四个方法中, LOF, LDOF 与 F-ABOD 三类均为孤立点检测方法; 而 OPTICS 在密度聚类上有较好的效果. 在二值检测中, 选择了两类密度聚类方法用来检测孤立点.

不排除三类数据集本身具有较大差异, 簇与簇之间的界限模糊的因素, 总体来说在非二值方法下, 随着参数 k 增大, 检测的 Top-10 结果具有上升或保持稳定的趋势. 其中, LOF 与 OPTICS 能检测出更多的点, 且预设孤立点排序和 ($SRoPO$) 较低, 说明其余未检测到的点的孤立程度也较接近最大的孤立值. LOFD 与 F-ABOD 分别是在基于距离与角度的因素下对孤立点进行判定, 检测的点数低于其他两类非二值检测方法. 其中基于距离的检测方法在近邻参数的调整下检测结果并无较大差异, $SRoPO$ 也并非随 k 的增加而减少; 这说明 1) 两类算法产生的 $\langle x, outlier-ness\rangle$ 结果对参数 k 略显敏感; 2) 检测结果与 k 难以成比例, 获取最佳结果需要不断调试参数试验. 而 Chess 数据集中, 数据的值分布仅为 1～7, 且均为整数; 因此, 孤立点的检测难度较大. 在非二值的四类方法下, 检测结果未及数据集 1、3 效果良好.

表 6　Re-Con DBSCAN 算法在模拟数据集 1～3 的检验结果

Table 6　Detection results of Re-Con DBSCAN on synthetic point sets 1～3

Parameter		Synthetic 1			Synthetic 2			Synthetic 3		
$\langle eps, eta\rangle$	$ToDR$	n_{DO}	HR	n_{DI}	n_{DO}	HR	n_{DI}	n_{DO}	HR	n_{DI}
$\langle 0.08, 0.011\rangle$	0.7	75	0.1412	–	79	0.1612	–	69	0.1324	–
	0.8	175	0.3296	1	163	0.3327	3	176	0.3378	2
	0.9	329	0.6196	1	256	0.5224	3	269	0.5163	–
	1	280	0.5273	–	226	0.4612	–	234	0.4491	–

表 7　Re-Con OPTICS 算法在模拟数据集 1～3 的检验结果

Table 7　Detection results of Re-Con OPTICS on synthetic point sets 1～3

Parameter		Synthetic 1			Synthetic 2			Synthetic 3		
$\langle eps, MinPts\rangle$	$ToDR$	n_{DO}	HR	n_{DI}	n_{DO}	HR	n_{DI}	n_{DO}	HR	n_{DI}
$\langle 0.08, 8\rangle$	10	345	0.6497	–	365	0.7449	–	345	0.6622	–
	30	450	0.8457	–	457	0.9327	–	460	0.8829	–
	50	475	0.8945	–	467	0.9531	–	481	0.9232	–
	70	476	0.8964	–	471	0.9612	–	490	0.9405	–

表 8 六种不同检测方法在三类公开数据集上的对比

Table 8 Detection results of six OD methods on three public higher dimensional datasets

数据集												
	非二值检测结果: TOP-n								二值检测结果			
k	LOF		LDOF		F-ABOD		OPTICS		DPC		ODIC-DBSCAN	
	Top 10	*SRoPO*	Top 10	*SRoPO*	Top 10	*SRoPO*	Top 10	*SRoPO*	*per*	*HR*	*para*	*HR*
Banknote Authentication												
5	6	160	2	2 291	2	3 326	8	154	1 %	6/10	2, 0.9	8/9
10	8	81	4	643	3	2 514	8	368	1.50 %	7/10	4, 0.9	8/9
15	7	217	4	485	3	1 962	8	89	2 %	8/10	5, 0.9	8/9
20	7	83	4	655	3	1 472	6	149	5 %	8/10	6, 0.9	8/9
Chess												
5	1	324	2	1 860	1	10 542	5	170	1 %	3/10	2, 0.9	10/136
10	4	308	2	521	–	10 019	1	243	1.50 %	3/10	2.5, 0.9	10/136
15	4	626	3	401	–	9 350	1	301	2 %	4/10	3, 0.9	10/136
20	3	765	2	447	–	8 326	1	394	5 %	4/10	3.5, 0.9	10/136
WDBC												
5	1	729	1	1 275	1	1 907	5	113	1 %	6/10	3, 0.9	5/6
10	6	76	4	378	1	1 673	5	145	1.50 %	6/10	4, 0.9	5/6
15	6	85	5	132	1	1 626	5	203	2 %	6/10	5, 0.9	5/6
20	5	91	6	112	3	1 459	5	233	5 %	6/10	6, 0.9	5/6

表 9 基于密度比的 ReCon-DBSCAN 与 ReCon-OPTICS 算法在三类数据集上的检测结果

Table 9 Detection results of density-ratio based ReCon-DBSCAN and ReCon-OPTICS on three real-world point sets

Parameter		数据集 1		数据集 2		数据集 3	
ToDR for RA-DBSCAN	*ToDR* for RA-OPTICS	ReCon-DBSCAN	ReCon-OPTICS	ReCon-DBSCAN	ReCon-OPTICS	ReCon-DBSCAN	ReCon-OPTICS
0.5	20 160 10	–	5	–	–	–	6
0.6	30 170 20	3/14	5	2/88	1	–	5
0.7	40 180 25	4/48	6	10/550	1	2/1	4
0.8	50 190 30	6/240	6	10/1 354	1	3/8	4
0.9	60 200 35	6/597	7	8/365	1	5/159	4
1	70 210 40	5/196	7	–	error	–	4

ODIC-DBSCAN 属于二值检测, 仅对数据对象判定是否为孤立点, 而非测定其孤立程度. 对此, 该方法显示出了较强的检测能力. 如数据集 1 中, 当有效点阈值在 5～6 时, 检测到的 9 个孤立点中有 8 个是预设的点. 在较难检测的数据集 2 中, 与其他几类算法结果不同之处是, ODIC-DBSCAN 算法检测出了所有预设孤立点, 但孤立点所在的簇中, 同样包含 126 个其他正常点. 这说明, 算法对孤立点的检测具有一定效果, 同时可检测出的其余点 (这些点可能是具有处于正反两类数据模糊边界的特性), ODIC-DBSCAN 善于捕捉类似的边界点, 正如算法对两个簇在融合之时能够检测出簇间点一般. 结合以上各类表的检测数据, 可以发现在真实数据集中, ODIC-DBSCAN 对簇间的孤立点检测依然有效果, 这是由于算法考虑了不同密度间的关联关系, 用不同的关键半径对数据集进行聚类, 因此算法普适性也较强.

4 结语

针对传统孤立点检测方法忽略簇内孤立点的问题, 提出了一种簇内孤立点检测技术 ODIC-DBSCAN. 该方法通过预处理工作与聚类算法的优化, 能够检测出预置的若干簇内孤立点. 算法首先依据数据集特性学习有效半径集合并构建密度矩阵; 然后提出点集覆盖模型, 利用拉格朗日乘子法求取极值, 输出点比阈值; 最后修改 DBSCAN 的核心对

象定义, 将预处理结果与优化 DBCSAN 算法相结合, 当簇发生了融合现象时, 输出孤立点检测结果. 下一步将在此思想的基础上在广度与深度上进一步提出其他优化的方法. 该优化包含两个方向: 1) 思想移植; 2) 效率优化. 在思想移植中, 工作目标致力于将自学习思想移植于其他相仿的聚类算法中, 使算法对孤立对象具备敏感性; 在效率优化问题中, 需要进一步将预处理思路与其他聚类算法特性相接轨, 在保证必备性能的基础上, 正确检测簇内孤立点, 使类似的检测方式更加活泛.

附录

算法 2. ODIC-DBSCAN

输入. 点集 $\boldsymbol{P}$, 距离矩阵 $\boldsymbol{dist}$, 点比阈值组 $\boldsymbol{GPRT}$, 半径集合 $Radius$

输出. $\boldsymbol{IDX}$

1) Function [$\boldsymbol{IDX}$, isnoise] = ODIC_DBSCAN($\boldsymbol{P}$, $\boldsymbol{dist}$, $\boldsymbol{GPRT}$, $Radius$)
2) for den_for = 1:(size($\boldsymbol{GPRT}$, 1)) /*标记 1*/
3) for $i = 1 : n$ /*点集中的 n 个点循环*/
4) if ˜visited(i) /*标记, 确认是否访问过*/
5) visited(i) = true;
6) $\boldsymbol{N}$ = QueryNeighbor(i, den_for);
/*子函数 3, 查询点 i 的邻居*/
7) $NRatio$ = QueryRatio(i, den_for);
/*子函数 2, 标记②*/
8) $C = C$+1; /*簇群加一*/
9) ExpandCluster(i, $\boldsymbol{N}$, C, den_for);
/*转至扩充区域的函数, 第 11 行*/
10) end if end for
11) end /*结束第二行的循环*/
12) function ExpandCluster(i, $\boldsymbol{N}$, C, den_for)
/*子函数 1: 该方法能扩充 i 所在的区域*/
13) while true
14) $j = \boldsymbol{N}(k)$;
15) if ˜visited(j)
16) visited(j) = true;
17) $\boldsymbol{N}2$ = QueryNeighbor(j, den_for);
/*返回 j 点的邻居*/
18) $JNRatio$ = QueryRatio(j, den_for);
/*返回 j 点当前的点数比*/
19) if $JNRatio \geq \boldsymbol{GPRT}(den\_for, 3)$
/*标记③*/
20) $\boldsymbol{N} = [\boldsymbol{N} \quad \boldsymbol{N}2]$; /*合并、扩充区域*/
21) end if
22) end if
23) if $\boldsymbol{IDX}(j) == 0$
24) $\boldsymbol{IDX}(j) = C$;
25) end end while
26) end /*子函数 1 END*/
27) function NR = QueryRatio(j, den_for)
/*子函数 2: 查询该点的点数比. 标记④*/
28) $\boldsymbol{N}1$=find($\boldsymbol{dist}(j,:) \leq Radius(den\_for + 1)$);
29) $\boldsymbol{N}2$=find($\boldsymbol{dist}(j,:) \leq Radius(den\_for)$);
30) $NeighborR$=numel($\boldsymbol{N}1$)./numel($\boldsymbol{N}2$);
31) end /*子函数 2 END*/
32) function $\boldsymbol{N}$ = QueryNeighbor(i,den_for)
/*子函数 3*/
33) $\boldsymbol{N}$ = find ($\boldsymbol{dist}(i, :) \leq Radius\ (den\_for)$);
34) end /*子函数 3 END*/
35) end of Function /*FUNCTION END*/

算法主要体现在四方面 (在附录处算法 2 中分别标记为①、②、③、④). 标记① (算法 2、11 行) 在 DBSCAN 的基础上构建了一个外层 for 循环, 该循环的变量为前述算法规定的点比阈值组, 即该点集中存在无差别曲线族的数目 (一般在 10 以内), 该循环的作用是渐进地逼近簇的最大化. 标记② (位于 27 ∼ 31 行) 为调用计算点数比的方法, 所调用的函数为 QueryRatio, 是标记④的内容, 计算了目标点不同半径下点数目的比值. 标记③ (位于 19 行) 修改了核心点的确认方法, 原始方法通过确认该点邻居数目是否大于规定 $MinPts$ 来判定是否将该点纳入扩充区域; 标记③通过记录点 j 的点数比与约定的点比阈值关系确认是否能将点 j 纳入簇的扩充区域, 修改了簇的扩充方法与原理.

算法 3. 簇群建立

输入. $\boldsymbol{IDX}$

输出. $\boldsymbol{ClusterFormate}$, $clusterNum$, $ClusterFormate$, $\boldsymbol{cluster\_Content\_return}$

% 输入中包括数据结构 $\boldsymbol{ClusterFormate}$

% 输出为数据结构 $\boldsymbol{ClusterFormate}$, 簇的数目 $clusterNum$, 簇的内容 $\boldsymbol{cluster\_Content\_return}$

1) Function [$\boldsymbol{ClusterFormate}$, $\boldsymbol{cluster\_Content\_return}$] = DBSCAN_ClusterFormate($\boldsymbol{IDX}$)
2) $\boldsymbol{IDX\_stats}$ = tabulate($\boldsymbol{IDX}$);
/*统计 $\boldsymbol{IDX}$ 下簇的分布情况*/
3) 计算每个簇最少点数, 记为 $ClusterThreshold$;
4) for $i \rightarrow \boldsymbol{IDX\_stats}$
/*for 循环确定是否能够形成核心簇*/
5) if $\boldsymbol{IDX\_stats}$ (当前簇内点数目比例) > $ClusterThreshold$
6) $\boldsymbol{Clusters}(i,:)$ = 添加该簇数据;
7) $i = i + 1$;
8) end if
9) end /*END FOR*/

10) if isempty($\boldsymbol{Clusters}$) /*若为空, 返回标记*/
11) $\boldsymbol{ClusterFormate} = [0\ 0\ 0]$;
12) else /*簇非空, 返回标记与簇内容*/
13) $\boldsymbol{ClusterFormate} = [\boldsymbol{Clusters}]$;
14) end if
15) if $(effectiveNum/\text{size}(\boldsymbol{P})) \geq EffectivePointsThreshold$
16) $\boldsymbol{ClusterFormate} = \boldsymbol{ClusterFormate}$

/*有簇分类号的点数目超过总体阈值*/

17) else
18) $\boldsymbol{ClusterFormate} = [0\ 0\ 0]$;

/*形成聚类的点的数目没有超过规定的阈值*/

19) end if
20) end of Function /* FUNCTION END*/

References

1 Yuan Chao, Chai Yi. Method for local community mining in the complex networks. *Acta Automatica Sinica*, 2014, **40**(05): 921−934
(袁超, 柴毅. 复杂网络的局部社团结构挖掘算法. 自动化学报, 2014, **40**(5): 921−934)

2 Yuan Hao-Nan, Guo Ge. Vehicle cooperative optimization scheduling in transportation cyber physical systems. *Acta Automatica Sinica*, 2019, **45**(1): 143−152
(原豪男, 郭戈. 交通信息物理系统中的车辆协同运行优化调度. 自动化学报, 2019, **45**(1): 143−152)

3 Wang Li, Zhang Rong, Sha Chao-Feng, Wang Xiao-Ling, Zhou Ao-Ying. A product normalization method for E-commerce. *Journal of Computers*, 2014, **37**(2): 312−325
(王立, 张蓉, 沙朝锋, 王晓玲, 周傲英. 电子商务商品归一化方法研究. 计算机学报, 2014, **37**(2): 312−325)

4 Hawkins D M. *Identification of outliers*. London: Chapman and Hall, 1980. 1

5 Zhang Y, Hamm N A S, Meratnia N, Stein A, Voort M, Havinga P J M. Statistics-based outlier detection for wireless sensor networks. *Int Journal of Geographical Information Science*, 2012, **26**(8): 1373−1392

6 Rousseeuw P J, Hubert M. Robust statistics for outlier detection. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2011, **1**(1): 73−79

7 Zhang K, Hutter M, Jin H. A new local distance-based outlier detection approach for scattered real-world data. *Advances in Knowledge Discovery and Data Mining*, 2009: 813−822

8 Wang B, Xiao G, Yu H, Yang X C. Distance-based outlier detection on uncertain data. In: Proceedings of the Ninth IEEE International Conference on Computer and Information Technology. Xiamen, China: IEEE, 2009. 293−298

9 Cassisi C, Ferro A, Giugno R, Pigola G, Pulvirenti A. Enhancing density-based clustering: Parameter reduction and outlier detection. *Information Systems*, 2013, **38**(3): 317−330

10 Tao Y, Pi D. Unifying density-based clustering and outlier detection. In: Proceedings of the Second International Workshop on Knowledge Discovery and Data Mining. Moscow, Russia: IEEE, 2009. 644−647

11 Gupta M, Gao J, Aggarwal C C, Han J W. Outlier detection for temporal data: A survey. *IEEE Trans on Knowledge and Data Engineering*, 2014, **26**(9): 2250−2267

12 Aggarwal C C, Yu P S. Outlier detection for high dimensional data. In: Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data. New York, USA: ACM, 2001. **30**(2): 37−46

13 Jiang B, Pei J. Outlier detection on uncertain data: objects, instances, and inferences. In: Proceedings of the 27th IEEE International Conference on Data Engineering. Hannover, Germany: IEEE, 2011. 422−433

14 Yu Yan-Wei, Wang Qin, Kuang Jun, He Jie. An on-line density-based clustering algorithm for spatial data stream. *Acta Automatica Sinica*, 2012, **38**(6): 1051−1059
(于彦伟, 王沁, 邝俊, 何杰. 一种基于密度的空间数据流在线聚类算法. 自动化学报, 2012, **38**(6): 1051−1059)

15 Branch J W, Giannella C, Szymanski B, Wolff R, Kargupta H. In-network outlier detection in wireless sensor networks. *Knowledge and Information Systems*, 2013, **34**(1): 23−54

16 Duan L, Xu L, Liu Y, Lee J. Cluster-based outlier detection. *Annals of Operations Research*, 2009, **168**(1): 151−168

17 Du H Z, Zhao S J, Zhang D Q, WU J S. Novel clustering-based approach for local outlier detection. In: Proceedings of the 2016 IEEE Conference on Computer Communications Workshops. CA, USA: IEEE, 2016. 802−811

18 Maheshwari K, Singh M. Outlier detection using divide-and-conquer strategy in density based clustering. In: Proceedings of the 2016 International Conference on Recent Advances and Innovations in Engineering. Jaipur, India: IEEE, 2016. 1−5

19 Breunig M M, Kriegel H P, Ng R T, Sander J. LOF: identifying density-based local outliers. In: Proceeding of the 2000 ACM SIGMOD International Conference on Management of Data. New York, USA: ACM, 2000. 93−104

20 Kriegel H P, Schubert M, Zimek A. Angle-based outlier detection in high-dimensional data. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Las Vegas, USA: ACM, 2008. 444−452

21 Rodriguez A, Laio A. Clustering by fast search and find of density peaks. *Science*, 2014, **344**(6191): 1492−1496

22 Wang X F, Xu Y. Fast clustering using adaptive density peak detection. *Statistical methods in medical research*, 2017, **26**(6): 2800−2811

23 Chu Rui-Hong, Wang Hong-jun, Yang Yan, Li Tian-Rui. Clustering ensemble based on density peaks. *Acta Automatica Sinica*, 2016, **42**(9): 1401−1412
(褚睿鸿, 王红军, 杨燕, 李天瑞. 基于密度峰值的聚类集成. 自动化学报, 2016, **42**(9): 1401−1412)

24 Bie R, Mehmood R, Ruan S, Sun Y C, Dawood H. Adaptive fuzzy clustering by fast search and find of density peaks. *Personal and Ubiquitous Computing*, 2016, **20**(5): 785−793

25 Wang M, Zuo W, Wang Y. An improved density peaks-based clustering method for social circle discovery in social networks. *Neurocomputing*, 2016, **179**: 219−227

26 Frey B J, Dueck D. Clustering by passing messages between data points. *Science*, 2007, **315**(5814): 972−976.

27 Givoni I, Frey B. Semi-supervised affinity propagation with instance-level constraints. In: Proceeding of the 2009 International Conference on Artificial Intelligence and Statistics. Florida, USA: 2009. 161−168

28 Inmar E, Givoni, Clement Chung, Brendan J. Frey. Hierarchical affinity propagation. In: Proceeding of the 27th Conference on Uncertainty in Artificial Intelligence. Barcelona, Spain: AUAI Press, 2011. 238−246

29 Ankerst M, Breunig M M, Kriegel H P, Sander J. OPTICS: ordering points to identify the clustering structure. In: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data. New York, USA: ACM, 1999. 49−60

30 Zhu Y, Ting K M, Carman M J. Density-ratio based clustering for discovering clusters with varying densities. *Pattern Recognition*, 2016, **60**: 983−997

31 Oh H K, Yoon S H, Kim S W. Hierarchical clustering and outlier detection for effective image data organization. In: Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication. New York, USA: ACM, 2012. Article No. 18

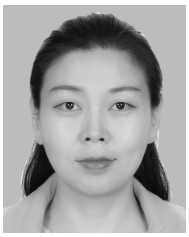
32 Kim K, Khabsa M, Giles C L. Inventor name disambiguation for a patent database using a random forest and dbscan. In: Proceedings of the 2016 IEEE/ACM Joint Conference on Digital Libraries. Newark, USA: IEEE, 2016. 269−270

33 Abid A, Kachouri A, Mahfoudhi A. Outlier detection for wireless sensor networks using density-based clustering approach. *IET Wireless Sensor Systems*, 2017, **7**(4): 83−90

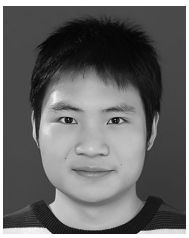
王跃飞 新疆大学信息科学与工程学院博士研究生. 主要研究方向为数据挖掘, 机器学习. 本文通信作者.
E-mail: yuefei_gezi@sina.com
(**WANG Yue-Fei** Ph. D. candidate at the College of Information Science and Engineering, Xinjiang University. His research interest covers data mining and machine learning. Corresponding author of this paper.)

于 炯 新疆大学信息科学与工程学院教授, 博士生导师. 主要研究方向为并行计算, 分布式系统, 绿色计算.
(**YU Jiong** Professor at the College of Information Science and Engineering, Xinjiang University, doctoral supervisor. His research interest covers parallel computing, distributed system and green computing.)

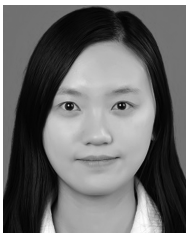
苏国平 新疆大学兼职教授, 博士生导师. 主要研究方向为计算机应用, 软件工程.
(**SU Guo-Ping** Professor at the Xinjiang University, doctoral supervisor. His research interest covers computer application and software engineering.)



钱育蓉 新疆大学软件学院教授, 博士生导师. 主要研究方向为大数据处理, 机器学习.
(**QIAN Yu-Rong** Professor at the School of Software, Xinjiang University, doctoral supervisor. Her research interest covers big data processing and machine learning.)



廖 彬 新疆财经大学副教授. 主要研究方向为分布式系统, 绿色计算.
(**LIAO Bin** Associate professor at the Xinjiang University of Finance and Economics. His research interest covers distributed system and green computing.)



刘 粟 新疆大学信息科学与工程学院硕士研究生. 主要研究方向为大数据处理, 机器学习.
(**LIU Su** Master student at the College of Information Science and Engineering, Xinjiang University. Her research interest covers big data processing and machine learning.)