软件学报 ISSN 1000-9825, CODEN RUXUEW
2025,36(2):851−873 [doi: 10.13328/j.cnki.jos.007177] [CSTR: 32375.14.jos.007177]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于密度的多度量空间数据聚类算法*

朱轶凡[1], 罗程阳[1], 马瑞遥[1], 陈 璐[1], 毛玉仁[2], 高云君[1]

[1](浙江大学 计算机科学与技术学院, 浙江 杭州 310027)

[2](浙江大学 软件学院, 浙江 宁波 315048)

通信作者: 毛玉仁, E-mail: yuren.mao@zju.edu.cn

摘 要: 具有噪声的基于密度的数据聚类 (DBSCAN) 算法是数据挖掘领域中的经典方法之一, 其不仅能发现数据中潜藏的复杂关系, 还能过滤其中的数据噪声, 从而获得高质量的数据聚类. 然而, 现有的基于密度的数据聚类算法仅支持单模态 (类型) 数据的聚类, 难以应对多模态 (类型) 数据并存的应用场景. 随着信息技术的快速发展, 数据呈现多模态化的发展态势, 现实生活中的数据不再是单一的数据类型, 而是多种数据模态 (类型) 的组合, 如文本、图像、地理坐标、数据特征等. 因此, 现有的数据聚类方法难以对复杂的多模态数据进行有效的数据建模, 更无法进行高效的多模态数据聚类. 基于此, 提出一种基于密度的多度量空间聚类算法. 首先, 为了刻画多模态数据间的复杂关系, 利用多度量空间表征数据之间的相似性关系, 并且利用聚合多度量图索引 (AMG) 实现多模态数据建模. 接着, 利用差分化的相似性关系优化聚合多度量图的图结构, 并且结合最优策略优先的搜索策略进行剪枝, 以实现高效的多模态数据聚类. 最后, 在真实与合成数据集上针对多种参数设置进行实验. 实验结果验证了所提方法运行效率提升了至少 1 个数量级, 并具有较高的聚类精度与良好的可扩展性.

关键词: 多度量空间; 多度量图; 基于密度的数据聚类; 数据挖掘; 多模态数据

中图法分类号: TP311

中文引用格式: 朱轶凡, 罗程阳, 马瑞遥, 陈璐, 毛玉仁, 高云君. 基于密度的多度量空间数据聚类算法. 软件学报, 2025, 36(2): 851–873. http://www.jos.org.cn/1000-9825/7177.htm

英文引用格式: Zhu YF, Luo CY, Ma RY, Chen L, Mao YR, Gao YJ. Density-based Data Clustering Algorithm in Multi-metric Spaces. Ruan Jian Xue Bao/Journal of Software, 2025, 36(2): 851–873 (in Chinese). http://www.jos.org.cn/1000-9825/7177.htm

Density-based Data Clustering Algorithm in Multi-metric Spaces

ZHU Yi-Fan[1], LUO Cheng-Yang[1], MA Rui-Yao[1], CHEN Lu[1], MAO Yu-Ren[2], GAO Yun-Jun[1]

[1](College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China)

[2](School of Software Technology, Zhejiang University, Ningbo 315048, China)

Abstract: The density-based spatial clustering of applications with noise (DBSCAN) algorithm is one of the clustering analysis methods in the field of data mining. It has a strong capability of discovering complex relationships between objects and is insensitive to noise data. However, existing DBSCAN methods only support the clustering of unimodal objects, struggling with applications involving multi-model data. With the rapid development of information technology, data has become increasingly diverse in real-life applications and contains a huge variety of models, such as text, images, geographical coordinates, and data features. Thus, existing clustering methods fail to effectively model complex multi-model data and cannot support efficient multi-model data clustering. To address these issues, in this study, a density-based clustering algorithm in multi-metric spaces is proposed. Firstly, to characterize the complex relationships within multi-model data, this study uses a multi-metric space to quantify the similarity between objects and employs aggregated multi-metric graph (AMG) to model multi-model data. Next, this study employs differential distances to balance the graph structure and leverages a best-first

* 基金项目: 国家重点研发计划 (2021YFC3300303); 国家自然科学基金 (62025206, 61972338, 62102351); 杭州市人工智能重大科技创新项目 (2022AIZD0116)
收稿时间: 2023-11-09; 修改时间: 2024-02-02; 采用时间: 2024-03-14; jos 在线出版时间: 2024-11-06
CNKI 网络首发时间: 2024-11-07

search strategy combined with pruning techniques to achieve efficient multi-model data clustering. The experimental evaluation on real and synthetic datasets, using various experimental settings, demonstrates that the proposed method achieves at least one order of magnitude improvement in efficiency with high clustering accuracy, and exhibits good scalability.

Key words: multi-metric space; multi-metric graph; density-based data clustering; data mining; multi-modal data

聚类是数据挖掘领域中的重要问题. 其依据数据的相似性特征将数据对象归类, 从而发现在数据中的潜藏模式和结构, 以对数据进行更深入的分析和理解. 基于密度的聚类算法 (density-based spatial clustering of applications with noise, DBSCAN)[1]是一种经典的聚类算法, 由于其在挖掘数据中潜藏结构信息的同时对数据的时间顺序不敏感, 且其能够过滤数据中的噪声信息, 其在时空数据管理[2,3]、流式数据挖掘[4,5]、多模态数据检索[6,7]、生物医疗[8,9]等领域具有广泛的应用场景. 然而, 现有的基于密度的聚类算法仅支持单模态 (类型) 的数据聚类, 无法应对多模态 (类型) 数据并存的应用场景. 随着信息技术的快速发展, 数据生态愈发完善, 数据类型愈发复杂. 现实生活中, 数据不再通过单一的模态表示. 相对的, 每个数据对象都是多种数据模态的组合, 如文本、图像、地理坐标、特征向量等. 由于现有基于密度的数据聚类算法难以对多模态数据进行有效的建模, 亦无法针对多模态数据进行高效的数据聚类. 因此, 面向多模态数据的基于密度聚类方面的研究尚属空白, 如何对结构复杂、类型多样的多模态数据进行高效的聚类分析仍是一个具有重要现实意义的开放问题.

多模态数据聚类能够实现更精准的数据聚类, 以满足用户的个性化需求. 图 1 给出了一个社交媒体应用中利用多模态数据聚类的示例. 在该社会关系网下, 每名用户具有多种类型的信息记录, 包括个人信息 (模态 A)、地理位置 (模态 B)、图片信息 (模态 C)、文本描述 (模态 D) 等. 这些数据之间蕴藏着大量的相似性特征, 包括单模态内部数据相似性 (如用户间的地理空间关系) 与跨模态数据相似性 (如同时考虑不同用户间的地理位置远近以及图片信息的相似性, 从而发现其中位置信息与图片信息的相似性). 然而, 现有的聚类方法大多针对某一固定的模态进行建模分析, 其局限于揭示单一模态的内部数据相似性特征. 相对而言, 通过多模态数据进行数据聚类可以将来自不同来源的数据进行融合 (如同时对模态 B、C、D 进行聚类), 以实现更全面的数据聚类, 从而揭示跨模态的数据相似性特征. 由于多模态数据聚类能为社交媒体、在线购物等应用场景下用户的个性化需求提供更优质、丰富的推荐服务, 因此多模态数据聚类分析任务具有重要的研究意义与广泛的应用前景.

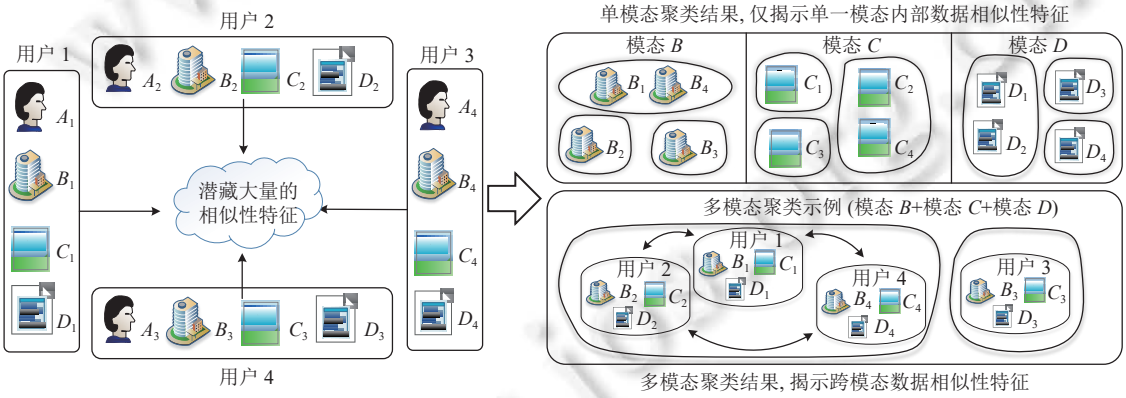


图 1 多模态数据聚类示例

针对多模态数据的建模与分析问题, 本文首先基于度量空间利用相似性度量方式将不同模态的数据进行通用地表征与管理, 从而建立通用的多模态数据表征模型. 具体而言, 给定一个度量方式 $d(\cdot)$, 对任意的数据对象 p_1 和 p_2, 其相似性距离可以表示为 $d(p_1, p_2)$. 由于度量空间不受数据模态 (类型) 特性的限制, 其可以处理任意的数据类型且支持灵活的距离度量方式. 因此, 近年来许多学者对基于密度的度量空间数据聚类方法展开研究, 如基于密度的分布式度量空间聚类[10], 异构图聚类[11,12]等. 然而, 对整体数据进行精确 DBSCAN 聚类的时间复杂度较高[13], 且其存在性能瓶颈[14]. 因此, 近年来众多学者基于近邻图对通用数据的 DBSCAN 展开研究. 其通过近邻图结构[15]、

近似度量策略[16]、中心近邻与边缘近邻结合策略[17]、传统索引与图索引结合[18]等方法在进行高效的数据聚类同时具有较高的聚类精度.

随着数据呈现多模态化的发展态势, 每个数据对象都是多种数据模态的组合. 由于现有的单度量空间下数据分析方法不支持同时管理多种多模态数据, 因此其难以满足实际应用中复杂的多模态数据分析需求. 基于此, 众多学者对多度量空间展开研究. 其同时将多个度量空间进行任意的线性组合形成独立的多度量空间, 并在多度量空间下对多模态数据建立统一的数据索引, 从而实现多种数据模态的同时管理与查询. 依据这些多度量空间索引对多模态数据的管理方式, 这些索引可以分为融合式多度量空间索引 (如 M^{2+}-tree[19]、M^3-tree[20]、RR*-tree[21]等) 与分离式多度量空间索引 (如 PM-tree[21], DESIRE[22]等). 在这些研究基础上, 一些学者对多度量空间分析展开研究, 如相似性检索[23]、相互最近邻[24]等, 然而, 这些索引并不能直接对多度量空间进行数据聚类.

研究多度量空间进行基于密度的数据聚类, 主要面临 3 大挑战.

(1) 如何灵活地表征多度量空间复杂的数据特性. 多度量空间下数据类型多样、结构复杂, 不同数据之间的特性存在较大差异. 现有的多度量空间索引通过融合式或者分离式索引对多度量空间进行管理, 然而融合式索引难以支持灵活的度量空间查询, 而分离式索引将数据分开存储造成较高的存储开销与额外的硬盘访问开销.

(2) 如何在多度量空间下进行高效的数据聚类. 基于密度的精确数据聚类方法需要对数据集中的每个数据对象进行相似性查询, 而后依据返回的结果进行数据聚类. 随着数据隐式维度愈发复杂, 其性能愈差[14]. 尽管一些学者对近似方法展开研究从而大幅提升效率, 但是现有方法难以同时处理不同模态的数据类型. 具体而言, 由于不同数据模态的特性不同, 现有方法难以对其建立统一的近邻关系图, 亦无法通过数据划分进行近似求和计算, 因此现有的基于密度的近似聚类方法难以处理复杂的多模态数据聚类.

(3) 如何实现高精度的多度量空间数据聚类. 在多度量空间中, 由于查询数据对象存在多种数据模态, 因此如何同时考虑不同的数据模态, 从而进行高精度的数据聚类是一大难题. 例如, 在某一模态下进行数据聚类的结果不一定是其他模态中的聚类结果, 而考虑所有模态得到的数据结果又不一定是待查询模态中的数据结果. 此外, 由于多模态数据间存在复杂的近邻关系, 且现有基于图的查询策略往往依循局部最优查询方法, 因此现有方法的查询精度较差.

针对上述挑战, 本文首先提出多度量空间下的聚合图索引实现数据的动态表征, 通过将数据之间的复杂关系表示为多度量空间下的聚合近邻关系, 从而实现多度量空间的有效表征. 接着, 为实现多度量空间下的高效聚类, 本文基于聚合多度量图提出高效的数据聚类方法, 通过多度量聚合图中的近邻关系进行高效的近似搜索, 并且利用基于三角不等式的剪枝策略突破因数据维度灾难导致的性能瓶颈, 进而实现多度量空间下的数据高效聚类. 而后, 为精确地在多度量空间内根据给定的任意多种查询模态进行数据聚类, 本文通过聚合多度量图构建与查询同时进行的方式进行在线聚类, 并且通过结果归并策略与差分化的近邻关系以提升多度量空间数据聚类的精确性.

本文工作的主要贡献可以总结为以下 4 点.

(1) 提出了一种基于密度的多度量空间数据聚类算法. 该算法通过构建聚合多度量图, 实现对多度量空间下多模态数据的高效近似聚类.

(2) 设计了一种聚合多度量图方法 (aggregated multi-metric graph, AMG). 该方法通过图结构与多模态数据间的近邻关系对多度量空间进行有效表征, 并通过聚合任意多种数据模态的内在关系以支持动态场景下多度量空间数据聚类, 以有效提升聚类精度.

(3) 提出了一种聚合多度量图构建与数据聚类同时进行的聚合多度量图构建方法. 该方法在图构建过程中保留差分化的距离关系, 从而有效提升近邻查询时返回结果的准确性, 并且保证多度量空间数据聚类的高效性与精确性.

(4) 在 3 个真实数据集与 1 个合成数据集上对所提出方法进行详尽的实验评估. 实验结果表明, 相较于两种基于现有多模态数据索引的 DBSCAN 方法, 本文提出的方法显著提升多度量空间数据聚类的效率, 且具有较高的精度与良好的可扩展性.

1 相关工作

本节介绍相关工作. 第 1.1 节回顾度量空间与多度量空间的相关工作, 第 1.2 节介绍基于密度的度量空间数据聚类相关工作.

1.1 度量空间与多度量空间

给定满足非负性、对称性、自反性、严格非负性、三角不等特性的距离度量方式, 度量空间将任意数据间的关系量化为度量方式计算得到的相似性距离, 因此度量空间支持任意的数据类型与灵活地度量方式. 然而, 由于度量空间内不存在维度信息, 因此需要通过高效的索引结构对其内部数据进行有效管理. 现有的度量空间索引可以分为基于支枢点的数据索引、基于数据划分的数据索引与基于混合方法的数据索引[25,26]. 其中基于支枢点的数据索引方法 (如 LAESA[27], MVPT[28]等) 预先选取多个支枢点, 并计算所有数据对象到所有支枢点的距离. 接着, 对这些距离建立欧氏空间的索引, 从而实现数据管理. 基于划分的索引方法 (如 LC[29], M-tree[30]等) 将度量空间划分为不同的紧密分区, 在查询处理的时候通过筛除不符合查询条件的空间来加速查询效率. 基于混合方法的数据索引 (如 GNAT[31], M-index[32], 支枢点索引树[33]等) 结合了基于划分的索引方法和基于支枢点的索引方法, 在建立索引时同时利用支枢点和空间划分方法对度量空间建立索引.

随着数据种类愈发复杂, 用户的查询需求呈现多样化的发展态势, 一些学者将多个度量空间进行融合形成多度量空间, 以在查询中同时考虑多种度量方式, 从而满足用户的个性化需求. 现有的多度量空间索引依据其对多个度量空间的管理方式可以分为融合式多度量空间索引与分离式多度量空间索引. 其中, 融合式多度量空间索引 (如 M^{2+}-tree[19]、M^3-tree[20]、RR*-tree[21]等) 将所有度量空间经过线性组合形成一个统一的多度量空间, 在查询时利用所有度量空间融合得到的距离关系进行剪枝, 因此在处理由少量度量空间融合得到的多度量空间查询时性能较差. 分离式多度量空间索引 (如 PM-tree[21], DESIRE[22]等) 将多个度量空间进行独立管理, 在查询时同时对多个索引进行独立查询, 而后通过查询结果融合方法将所有的结果进行筛选得到最终结果. 尽管该方法较为灵活, 但是其在处理查询时需要对多个索引进行独立查询, 因此查询性能依然存在瓶颈.

1.2 基于密度的度量空间数据聚类

基于密度的度量空间数据聚类方法可以分为两类: 精确聚类方法与近似聚类方法. 精确的聚类方法基于现有的度量空间索引框架, 实现基于密度的度量空间数据聚类, 如基于密度的分布式度量空间聚类[10]方法在 KD-tree[34]基础上将度量空间进行空间划分, 而后在分布式环境下利用现有聚类框架实现度量空间数据聚类; 异构图聚类[11,12]方法将数据通过随机游走的方式构建相似性多属性图, 而后在图中进行遍历方法, 从而得到基于密度的精确数据聚类结果; 分布式空间索引 PDBSCAN[35]利用分布式环境进行并行 R*-tree 构建从而实现空间聚类. 近似的聚类方法通过构建近邻图关系将相似的数据进行关联, 而后在图中进行遍历得到近似的数据聚类, 如近邻图方法[15]直接对所有数据构建近邻图, 而后通过遍历返回数据聚类; 近似度量策略法[16]在构建近邻图时同时考虑近似数据之间的相似性关系以进一步提升效率. 然而, 近似方法存在精度问题, 即图中近邻关系难以保证在进行密度较为稀疏的数据聚类时的结果精度. 因此, 数据中心近邻与边缘近邻结合法[17]在构建图时同时考虑长距离边 (即不相似数据关系) 与短距离边 (即相似数据关系) 从而提升聚类精度, 而传统索引与图索引结合[18]等方法通过在传统索引基础上构建近邻图的方法, 同时进行全局精确查询与局部近似搜索从而提升基于密度的数据聚类精度. 然而, 上述方法均是对于单度量空间的数据聚类, 因此难以扩展至多度量空间下以满足用户的复杂个性化需求.

2 基础知识

本节介绍本文工作的基础知识. 第 2.1 节介绍度量空间与多度量空间的相关知识, 第 2.2 节介绍度量空间与多度量空间下基于密度数据聚类的基本概念.

2.1 度量空间与多度量空间

度量空间可以由形如 (S,d) 的二元组形式表示, 其中 $S=\{p_1,p_2,\ldots,p_n\}$ 是数据对象的集合, 而 $d:S\times S\to R$ 是

给定的度量函数. 在度量空间 (S,d) 下, 其中的数据对象 $p\in S$ 可以是任意的数据类型, 如数值、文本、图片、高维向量等, 而其中的距离度量方式 $d(\cdot,\cdot)$ 将数据对象间的相似性量化, 并且需要满足以下 5 个特性.

(1) 非负性

$$\forall p_1,p_2\in S,\ d(p_1,p_2)\geqslant 0 \tag{1}$$

(2) 对称性

$$\forall p_1,p_2\in S,\ d(p_1,p_2)=d(p_2,p_1) \tag{2}$$

(3) 自反性

$$\forall p_1\in S,\ d(p_1,p_1)=0 \tag{3}$$

(4) 严格非负性

$$\forall p_1,p_2\in S,\ p_1\neq p_2\Rightarrow d(p_1,p_2)>0 \tag{4}$$

(5) 三角不等特性

$$\forall p_1,p_2,p_3\in S,\ d(p_1,p_3)\leqslant d(p_1,p_2)+d(p_2,p_3) \tag{5}$$

基于一个满足以上 5 个特性的距离度量方式 $d(\cdot,\cdot)$, 度量空间将数据对象间的相似性距离量化, 因而可以对任意类型的数据建模. 然而, 随着数据模态愈发繁杂, 用户的个性化需求开始考虑多种数据模态, 单一的数据模态难以满足实际的用户需求. 因此, 需要将多个度量空间融合为多度量空间以同时表征多种数据模态.

基于度量空间的定义形式, 多度量空间可以由形如 $(\overline{S},\overline{d})$ 的二元组形式表示, 其中 $\overline{S}=\{\overline{p}_1,\overline{p}_2,\ldots,\overline{p}_n\}$ 是多度量数据对象的集合, $\overline{d}=\{d_1,d_2,\ldots,d_m\}$ 是多种度量函数的集合. 在多度量空间下, 每一个多度量数据对象都是多个单度量数据对象的组合, 例如多度量数据对象 $\overline{p}_1=\{p_1^1,p_1^2,\ldots,p_1^m\}$, 其中 p_1^i 表示 $\overline{p}_1$ 在第 i 个度量空间中的数据. 此外, 多度量空间支持任意多个度量方式的组合. 具体而言, 给定一个权重向量集 $\overline{W}=\{w_1,w_2,\ldots,w_m\}$, $w_i\in[0,1]$ $(1\leqslant i\leqslant m)$, 多度量空间中任意两个多度量数据对象 $\overline{p}_1$ 和 $\overline{p}_2$ 的距离可以由 $\overline{d}(\overline{p}_1,\overline{p}_2)=\Sigma_{1\leqslant i\leqslant m}w_i\times d_i(\overline{p}_1,\overline{p}_2)$ 表示. 注意到, 多度量空间中的权重向量集 $\overline{W}$ 可以由用户自定义, 因此可以支持对任意多种数据模态进行动态加权的数据表征. 然而, 多度量空间中不存在维度特征 (即, 多度量空间内的数据对象无法依据具体的维度进行空间划分), 这导致多度量空间在查询时仅支持对数据相似性特征进行查询操作, 如多度量 k-最近邻查询.

定义 1. 多度量 k-最近邻查询. 给定一个多度量空间 $(\overline{S},\overline{d})$, 一个查询数据对象 $\overline{q}$, 一个正整数 k 以及一个查询权重的集合 $\overline{W}=\{w_1,w_2,\ldots,w_m\},w_i\in[0,1]$ $(1\leqslant i\leqslant m)$, 多度量 k-最近邻查询找到该多度量空间下与查询对象最为相似的 k 个数据对象, 即 $MkNNQ(\overline{q},k)=\{\overline{S}'\mid\overline{S}'\subseteq\overline{S}\wedge|\overline{S}'|=k\wedge\forall\overline{p}\in\overline{S}',\forall\overline{p}'\in\overline{S}-\overline{S}',\overline{d}(\overline{q},\overline{p})\leqslant\overline{d}(\overline{q},\overline{p}')\}$.

2.2 基于密度的数据聚类

给定一个多度量空间 $(\overline{S},\overline{d})$, 基于密度的数据聚类方法 (DBSCAN)[1]基于近邻距离限制参数 ϵ 与近邻数量限制参数 $minpts$ 定义了一个基于密度的数据聚类. 其核心思想是将所有多度量数据对象依据稠密性 (即近邻数量) 进行区域划分, 并且将其中稠密的区域进行关联, 将稀疏的区域进行分裂. 最终, 划分至同 $\overline{q}$ 一个区域内的多度量数据对象即属于同一个数据聚类.

定义 2. 多度量 ϵ-最近邻. 给定一个多度量空间 $(\overline{S},\overline{d})$, 一个近邻距离限制参数 ϵ, 一个多度量数据对象 $\overline{p}$ 的 ϵ-近邻 $N_\epsilon(\overline{p})$ 包括所有距离 $\overline{p}$ 不超过距离限制 ϵ 的数据对象, 即 $N_\epsilon(\overline{p})=\{\overline{p}'\mid d(\overline{p},\overline{p}')\leqslant\epsilon\}$.

定义 3. 核数据对象. 给定一个多度量空间 $(\overline{S},\overline{d})$, 一个近邻数量限制参数 $minpts$, 若多度量数据对象 $\overline{c}$ 满足条件 $N_\epsilon(\overline{c})\geqslant minpts$, 则称该对象为核数据对象. 相对的, 若一个多度量数据对象 $\overline{p}$ 不满足条件 $N_\epsilon(\overline{p})\geqslant minpts$, 则称该多度量数据对象是一个非核数据对象.

定义 4. 基于密度的直接可达性. 给定一个多度量空间核数据对象 $\overline{c}$, 一个多度量空间数据对象 $\overline{p}$ 以及一个近邻距离限制参数 ϵ. 若 $\overline{p}\in N_\epsilon(\overline{c})$, 则称核数据对象 $\overline{c}$ 基于密度直接可达 $\overline{p}$. 注意到, 若 $\overline{p}$ 也是核数据对象, 则 $\overline{p}$ 和 $\overline{c}$ 的直接可达性具有自反性, 即 $\overline{p}$ 基于密度直接可达 $\overline{c}$. 反之, 若 $\overline{p}$ 不是核数据对象, 则 $\overline{p}$ 基于密度不能直接可达 $\overline{c}$.

定义 5. 基于密度的可达性. 给定一个多度量空间核数据对象 $\overline{c}$, 一个多度量数据对象 $\overline{p}$ 以及一个近邻距离限

制参数 ϵ. 若存在一个多度量数据对象序列 $\overline{P}=\{\overline{p}_1,\overline{p}_2,\ldots,\overline{p}_l\}$, 其中 $\overline{p}_1=\overline{c}$ 且 $\overline{p}_l=\overline{p}$, 并且该序列中的任意两个多度量数据对象 $\overline{p}_i,\overline{p}_{i+1}$ $(1\leqslant i\leqslant l)$ 均满足 $\overline{p}_{i+1}\in N_\epsilon(\overline{p}_i)$ (即 $\overline{p}_i$ 基于密度直接可达 $\overline{p}_{i+1}$), 则称 $\overline{c}$ 基于密度可达 $\overline{p}$.

定义 6. 基于密度的联通性. 给定两个多度量空间数据对象 $\overline{p}_1$ 和 $\overline{p}_2$, 若存在一个多度量数据对象 $\overline{p}$ 使得 $\overline{p}$ 基于密度可达 $\overline{p}_1$ 且 $\overline{p}$ 基于密度可达 $\overline{p}_2$, 则称 $\overline{p}_1$ 和 $\overline{p}_2$ 基于密度联通.

定义 7. 基于密度的数据聚类. 一个多度量空间 $\overline{S}$ 的子集 $C\in\overline{S}$ 是一个基于密度的聚类, 当且仅当 (1) 存在一个多度量空间数据对象 $\overline{p}\in C$, $\overline{p}$ 是一个核数据对象; (2) 对于任意两个多度量数据对象 $\overline{p}_1,\overline{p}_2\in C$, $\overline{p}_1$ 和 $\overline{p}_2$ 基于密度联通; (3) 不存在多度量数据对象 $\overline{p}_1\in C$ 和 $\overline{p}_3\notin C$, 使得 $\overline{p}_1$ 和 $\overline{p}_3$ 基于密度联通.

基于定义 7, DBSCAN 利用核数据对象将多度量空间划分至若干个稠密区域, 而后将这些稠密区域基于核数据对象间的联通性进行连接, 从而实现数据聚类. 图 2 给出一个 DBSCAN 聚类的例子, 其中包括一个多度量空间核数据对象集 $\{\overline{c}_1,\overline{c}_2,\ldots,\overline{c}_{10}\}$ 与多度量空间非核数据对象 $\{\overline{p}_1,\overline{p}_2,\ldots,\overline{p}_{13}\}$. 根据定义 7, 对该多度量空间中的数据对象进行基于密度的数据聚类后得到 3 个聚类: $\overline{C}_1=\{\overline{c}_1,\overline{c}_2,\ldots,\overline{c}_6,\overline{p}_1,\overline{p}_2,\ldots,\overline{p}_7\}$, $\overline{C}_2=\{\overline{c}_7,\overline{c}_8,\overline{c}_9,\overline{p}_8,\overline{p}_9,\ldots,\overline{p}_{12}\}$ 以及由一个核数据对象组成的聚类 $\overline{C}_3=\{\overline{c}_{10},\overline{p}_{12},\overline{p}_{13}\}$. 注意到, 其中虽然数据对象 $\overline{p}_{12}$ 和聚类 $\overline{C}_2$、$\overline{C}_3$ 中的数据对象均存在联通性, 但是由于聚类 $\overline{C}_2$ 中的数据对象和聚类 $\overline{C}_3$ 中的数据对象不存在联通关系 (例如 $\overline{c}_9$ 和 $\overline{c}_{10}$ 不存在基于密度的可达关系), 因此聚类 $\overline{C}_2$ 和 $\overline{C}_3$ 不能合并从而形成一个更大规模的聚类.

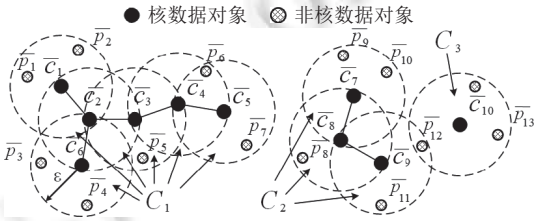


图 2 DBSCAN 示例

3 基于聚合多度量图的多度量空间数据聚类

本节介绍多度量空间数据聚类方法. 第 3.1 节介绍多度量空间数据聚类框架, 第 3.2 节介绍聚合多度量图的构建技术细节. 最后第 3.3 节提出基于聚合多度量图的多度量空间数据聚类方法, 从而在有限的精度损失下实现高效的多度量空间数据聚类.

3.1 多度量空间数据聚类

根据多度量空间 DBSCAN 的定义, 在多度量空间内进行基于密度的数据聚类包括两个步骤, 即根据给定的查询参数 (距离限制参数 ϵ 与规模限制参数 $minpts$) 找到所有的核数据对象, 以及将所有的核数据对象通过基于密度的联通性进行聚合从而形成基于密度的数据聚类. 因此, 可以直接得到多度量空间 DBSCAN 的算法框架 (如算法 1 所示).

算法 1. 多度量空间 DBSCAN 算法框架.

输入: 多度量空间 $(\overline{S},\overline{d})$, 近邻距离限制参数 ϵ, 近邻数量限制参数 $minpts$;
输出: 多度量空间内所有基于密度的数据聚类.

1. **for each** $\overline{p}\in\overline{S}$ **do**
2. 计算 $N_\epsilon(\overline{p})$
3. **if** $N_\epsilon(\overline{p})\geqslant minpts$ **then** /*该数据对象为核数据对象*/
4. 标记 $\overline{p}$ 为核数据对象
5. **for each** 核数据对象 $\overline{c}\in\overline{S}$ **do**

6. $C_{\bar{c}} \leftarrow |\bar{c}|$
7. **for each** 核数据对象 $\bar{p} \in N_{\epsilon}(\bar{c})$ **do**
8. **if** $|N_{\epsilon}(\bar{p})| \geqslant minpts$ **then** /* $\bar{c}$ 的多度量 ϵ-近邻 $\bar{p}$ 也是核数据对象*/
9. $C_{\bar{p}} \leftarrow \bar{p}$ 对应的聚类
10. $C_{\bar{c}} \leftarrow C_{\bar{c}} \cup C_{\bar{p}}$
11. **else**
12. $C_{\bar{c}} \leftarrow C_{\bar{c}} \cup \bar{p}$
13. $C \leftarrow \varnothing$
14. **for each** 核数据对象 $\bar{c} \in \overline{S}$ **do**
15. $C \leftarrow C \cup C_{\bar{c}}$
16. **return** 所有找到的多度量空间聚类 C

算法 1 的输入为多度量空间 $(\overline{S},\overline{d})$, 近邻距离限制参数 ϵ, 近邻数量限制参数 $minpts$. 算法首先通过现有的多度量空间索引找到空间内的所有数据对象的多度量 ϵ-近邻, 而后根据近邻数量限制 $minpts$ 判断其中的数据对象是否为核数据对象 (第 1–4 行). 接着, 算法将所有的核数据对象进行联通从而找到所有数据聚类. 具体来说, 对于每个核数据对象 $\bar{c} \in \overline{S}$, 算法判断 $\bar{c}$ 的每个多度量 ϵ-近邻是否也是核数据对象. 若 $\bar{p}$ 也是核数据对象, 则将 $\bar{c}$ 所在的聚类与 $\bar{p}$ 所在的聚类合并 (第 8–10 行); 否则, 即 $\bar{p}$ 不是核数据对象, 则将 $\bar{p}$ 合并至 $\bar{c}$ 所在的聚类中 (第 11,12 行). 最终, 算法返回所有找到的不重复聚类的集合 C (第 13–16 行).

值得注意的是, 在数据规模为 n, 数据维度为 λ 的单度量空间中进行 DBSCAN 的时间复杂度存在下限 $O(n^{2-\frac{2}{1+\lambda/2}})$ [14]. 因此, 在算法 1 中, 在进行第 1–4 行的时间复杂度不低于 $O(n^{2-\frac{2}{1+\lambda/2}})$. 而在进行第 5–11 行中的核数据对象连接时, 可以通过并查集的方法进行数据联接, 因此其时间复杂度为 $O\left(\sum |N_{\epsilon}(\bar{c})|\right)$, $\bar{c} \in \overline{S}$. 由于该数量级等同于在多度量空间下核数据对象的所有 ϵ-近邻数量, 因此算法 1 的效率瓶颈在于第 1–4 行中计算所有多度量数据对象的 ϵ-近邻. 在实际应用场景下, 即使面临 10 万级别的数据规模时, 通过算法 1 计算基于密度的多度量数据聚类将面临数小时的高昂时间开销. 因此, 亟需高效的多度量空间 DBSCAN 方法, 以应对实际应用中的大规模数据聚类需求.

考虑到现有的近似方法通过近邻图关系对度量空间内的数据对象相似度进行表征, 从而实现高效的 DBSCAN 聚类. 并且, 现有的多度量空间数据查询方法大多是基于多个模态的聚合方法. 因此, 一种有效的多度量空间 DBSCAN 聚类方法可以考虑聚合多个度量近邻图以表征每种数据模态下的相似性关系, 在进行多度量空间 DBSCAN 时将待查询的模态所对应的度量近邻图进行动态聚合, 而后进行图遍历查询. 由于基于度量近邻图的相似查询算法相比基于传统树形索引的聚类算法, 其性能更高[36], 因此能够更为明显地提升多度量空间 DBSCAN 聚类的性能.

后文图 3 给出了一个聚合多度量图 (aggregated multi-metric graph, AMG) 的例子, 其中展示了如何针对多度量空间 $\overline{S} = \{S_1, S_2\}$ 下的数据对象集 $\{\bar{p}_1, \bar{p}_2, \bar{p}_3, \bar{p}_4\}$ 建立聚合多度量图 $\overline{G}$ 的具体步骤. 首先, 针对度量空间 S_1 与 S_2, 对其中数据对象依据相似性关系建立度量近邻图 G_1 与 G_2. 例如, 对于数据对象 $\bar{p}_1$ 而言, 其在第 1 个度量空间 S_1 中的数据部分 p_1^1 与该空间中的数据对象 p_3^1 和 p_4^1 存在近邻关系, 因此将其相连. 同理, 将 $\bar{p}_1$ 在第 2 个度量空间 S_2 中的数据部分 p_1^2 与空间 S_2 中的数据对象 p_2^2 和 p_4^2 相连. 基于此, 在聚合多度量图 $\overline{G}$ 中, 将 $\bar{p}_1$ 与度量近邻图 G_1 与 G_2 中所有和 $\bar{p}_1$ 存在近邻关系的数据对象相连, 因此 $\bar{p}_1$ 在聚合多度量图中与 $\bar{p}_2$, $\bar{p}_3$ 和 $\bar{p}_4$ 相连.

3.2 聚合多度量图构建

基于聚合多度量图, 算法 1 中寻找所有数据对象的多度量 ϵ-近邻步骤转变为在每个度量空间下寻找数据对象的度量 ϵ-近邻后, 将所有度量空间中的结果汇总. 然而, 在每个度量空间中进行数据相似性查询后进行结果聚合时, 会存在较大的精度误差. 图 4 给出了一个聚合多度量图的查询示例. 其中, 针对多度量空间中的数据对象

$\{\overline{p}_1,\overline{p}_2,\overline{p}_3,\overline{p}_4\}$ 在每个度量空间下建立 k=1 的近邻图 (即, 每个数据对象仅与其最近的 k 个数据对象相连), 因此得到度量空间 S_1 中的近邻图 G_1 与度量空间 S_2 中的近邻图 G_2. 假设以多度量空间数据对象 $\overline{p}_1$ 为查询对象进行 $\epsilon=4$ 的近邻查询, 且在每个度量空间中的相似性度量方式均为 L_1 距离 (即 $d(p,q)=|p-q|$), 而该多度量空间内的距离度量方式为 $\overline{d}(\overline{p},\overline{q})=d^1(p^1,q^1)/2+d^2(p^2,q^2)/2$. 其首先在每个多度量空间中根据近邻关系找到 $\overline{p}_1$ 在每个度量空间中的 ϵ-近邻, 在度量空间 S_1 中查询并返回结果 $\{\overline{p}_1,\overline{p}_2\}$, 在度量空间 S_2 中查询并返回结果 $\{\overline{p}_1,\overline{p}_3\}$. 接着, 将所有待查询度量空间中的结果进行验证与汇总, 得到结果 $\{\overline{p}_1,\overline{p}_2,\overline{p}_3\}$. 然而, 在该多度量空间中进行 $\epsilon=4$ 的近邻查询的正确结果应该是 $\{\overline{p}_1,\overline{p}_2,\overline{p}_3,\overline{p}_4\}$. 因为数据对象 $\overline{p}_4$ 在每个独立的度量空间中均不满足 k=1 的近邻条件 (即 $\overline{p}_4$ 并不是 $\overline{p}_1$ 在每个度量空间中的最近邻), 但是其在给定的多度量查询空间中满足 $\overline{d}(\overline{q}=\overline{p}_1,\overline{p}_4)=d^1(p_1^1,p_4^1)/2+d^2(p_1^2,p_4^2)/2\leqslant\epsilon$. 因此, 将现有的单度量空间下基于近邻图的 DBSCAN 方法直接运用于多度量空间下虽然一定程度上提升查询效率, 但是其得到的查询结果精度较差, 难以满足实际应用中的用户需求.

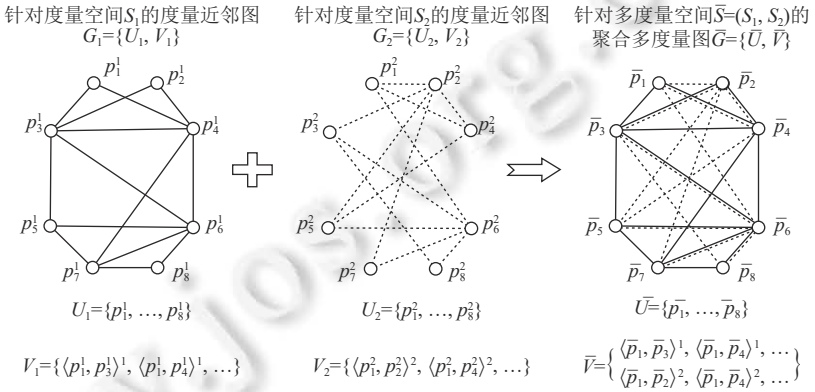


图 3 聚合多度量图示例

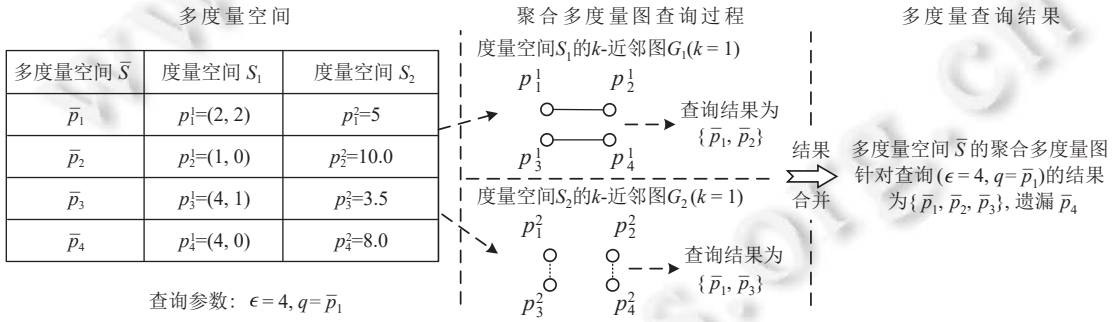


图 4 聚合多度量图查询示例

在图 4 的例子中, 注意到其中每个度量空间中的近邻图都是不联通的. 例如, 在度量空间 S_1 对应的近邻图 G_1 中, 数据对象 p_1^1 和 p_3^1 是不联通的, 即二者无法通过现有图上的边访问对方. 因此, 为解决图上的联通性问题, 本文采用迭代联通的方式, 将每个数据对象依次合并入现有的近邻图中. 初始状态下近邻图仅为第 1 个数据对象, 而后迭代地加入数据对象. 由于新加入的数据对象必然和先前加入的数据对象联通, 因此每次插入一个新的数据对象后近邻图内的所有数据对象依然是互相联通的.

此外, 在进行基于近邻图的 DBSCAN 时, 需要对所有的数据对象进行 ϵ-近邻查询, 其时间复杂度为 O(n·单次 ϵ-近邻查询复杂度). 然而, 在进行度量近邻图的构建时, 其对每个数据对象寻找 k-近邻, 其时间复杂度亦是 O(n·单次 k-近邻查询复杂度). 考虑到 k-近邻查询返回至多 k 个数据对象, 因此在实际查询中 ϵ-近邻查询返回的数据规模和 k-近邻查询的结果规模属于同一个数量级. 因而, 在进行 DBSCAN 查询时重构近邻图将不影响总查询的时间复

杂度. 基于此, 本文提出在进行多度量近邻查询时, 重构基于查询度量空间的近邻图, 以提升查询精度 (即, 不同于图 4 中对每个独立的度量空间构建近邻图, 该方法直接构建基于待查询多度量空间的近邻图). 由于该策略仅构建单个基于待查询的多度量空间的近邻图, 因此其查询效率与待查询的度量空间个数无关, 因此也提升了查询效率.

基于该策略, 图 5 给出针对图 4 中 ϵ-近邻问题构建的一个聚合多度量图示例. 其中, 初始状态下图中仅有一个多度量数据对象 $\overline{p}_1$. 接着, 插入数据对象 $\overline{p}_2$, 并且在现有图中找到 $\overline{p}_2$ 的所有 k-近邻并且进行联接. 而后以此类推, 加入数据对象 $\overline{p}_3$ 和 $\overline{p}_4$. 最终, 所有多度量数据对象均加入图中, 并且图所有的数据对象均是互相联通的. 此外, 由于图中所有的数据对象都是依据图中现有的数据对象进行近邻关系联接, 因此图中的联接关系不是严格的近邻关系. 不仅如此, 在该策略下图中会存在差分化的边权关系, 即先加入的边可能存在长距离边. 在查询时, 这些长距离边将用来扩张查询范围, 从而搜索到相比 k-近邻关系更大的范围区间, 以进一步提升 ϵ-近邻搜索精度. 不仅如此, 由于长距离边的存在, 在多度量图中进行遍历查询时能够更快地访问所需要查询的位置, 因此进一步地提升了查询效率.

多度量空间

多度量空间 $\overline{S}$	度量空间 S_1	度量空间 S_2
$\overline{p}_1$	p_1^1=(2, 2)	p_1^2=5
$\overline{p}_2$	p_2^1=(1, 0)	p_2^2=10.0
$\overline{p}_3$	p_3^1=(4, 1)	p_3^2=3.5
$\overline{p}_4$	p_4^1=(4, 0)	p_4^2=8.0

聚合多度量图构建

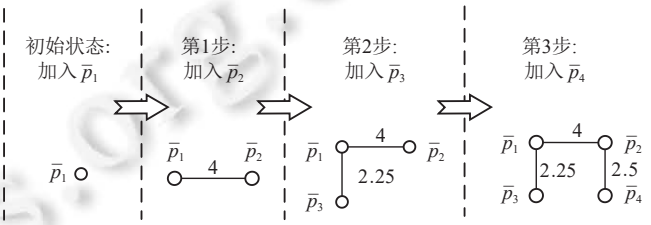


图 5 聚合多度量图构建示例

算法 2 描绘了聚合多度量图的构建算法, 其输入为多度量空间 $(\overline{S},\overline{d})$, 近邻参数 k, 以及待查询的多度量距离函数 $\overline{d}(\cdot,\cdot)$. 算法首先初始化聚合多度量图, 并且将多度量空间中的第 1 个数据对象加入图中. 接着, 其找到空间内的剩下所有数据对象的 k-近邻并且建立近邻关系. 具体而言, 对于每一个多度量数据对象 $\overline{p}$, 其首先初始化一个优先队列 *que* (队列内部根据数据对象到 $\overline{p}$ 的距离进行排序, 每次弹出距离 $\overline{p}$ 最近的数据对象), 一个结果集 *ans* 用于存放 $\overline{p}$ 的所有 k-近邻数据对象, 以及一个局部距离最值 dis_k 用于保存当前访问过的数据对象中距离 $\overline{p}$ 第 k 远的距离 (第 4–6 行). 接着, 其执行 while 循环, 直到队列 *que* 为空. 在每一次循环迭代中, 对于当前的队首元素 u (第 8, 9 行), 其找到图中所有与 u 存在近邻关系的数据对象 v (第 10 行). 并且, 若 v 满足 $\overline{d}(v,\overline{p}) < dis_k$ (即 v 可能是一个 $\overline{p}$ 的 k-近邻), 则用 v 更新结果候选集 *ans*, 局部距离最值 dis_k 以及优先队列 *que* (第 11–14 行). 最终, 当队列 *que* 为空时, 算法将所有结果集 *ans* 中的数据对象与 $\overline{p}$ 建立近邻关系. 当所有数据对象均被加入图中时, 算法返回由所有多度量数据对象与其内部近邻关系组成的聚合多度量图 $\overline{G}$ (第 16 行).

算法 2. 聚合多度量图构建算法.

输入: 待查询的多度量空间 $(\overline{S},\overline{d})$, 近邻参数 k;
输出: 聚合多度量图索引 $\overline{G}$.

1. $\overline{p}_{\text{ini}} \leftarrow \overline{S}$ 中的第 1 个数据对象
2. $\overline{G} \leftarrow \overline{G} \cup \{\overline{p}_{\text{ini}}\}$
3. **for each** $\overline{p} \in \overline{S}$ 且 $\overline{p} \neq \overline{p}_{\text{ini}}$ **do**
4. *que* $\leftarrow \{\overline{p}_{\text{ini}}\}$ /*初始化优先队列 *que* */
5. *ans* $\leftarrow \{\overline{p}_{\text{ini}}\}$ /*初始化结果集 *ans* 用于存放 $\overline{p}$ 的所有 k-近邻数据对象*/
6. $dis_k \leftarrow \infty$ /*当前访问过的数据对象中距离 $\overline{p}$ 第 k 远的数据对象到 $\overline{p}$ 的距离, 即 *ans* 中距离 $\overline{p}$ 最远数据对象的距离)*/
7. **while** *que* $\neq \varnothing$ **do**

8. $u \leftarrow que.top()$ /*找到优先队列 que 中的第 1 个元素*/
9. $que.pop()$
10. **for each** v, v 是 u 的 k-近邻 **do**
11. **if** $\overline{d}(v,\overline{p}) < dis_k$ **then**
12. 用 v 更新 ans
13. 用 $\overline{d}(v,\overline{p})$ 更新 dis_k
14. $que \leftarrow que \cup \{v\}$
15. 将所有 ans 中的数据对象与 $\overline{p}$ 建立近邻关系并且加入 $\overline{G}$ 中
16. **return** 所有多度量数据对象与其内部近邻关系组成的聚合多度量图 $\overline{G}$

3.3 基于聚合多度量图的多度量空间数据聚类

尽管聚合多度量图通过近邻关系实现多度量空间的动态高效表征, 但是现有的近邻图方法仅支持最近邻查询. 即, 在进行多度量空间 DBSCAN 时, 需要考虑每个数据对象在以查询半径为近邻距离限制参数 ϵ 的多度量 ϵ-近邻. 因此, 需要在现有的多度量近邻查询 (即算法 2 中第 4–14 行) 基础上进行多度量 ϵ-近邻查询. 一种直观可行的查询策略是在每次基于当前访问的数据对象进行向外扩张时, 对距离限制参数进行调整. 相较于现有算法中根据局部距离最值 dis_k 作为是否进行扩张的判别条件, 多度量 ϵ-近邻查询将近邻距离限制参数 ϵ 作为限制条件. 因此, 可以得到算法 3.

算法 3. 基于聚合多度量图的多度量 ϵ-近邻查询算法.

输入: 聚合多度量图索引 $\overline{G}$, 近邻距离限制参数 ϵ, 查询数据对象 $\overline{q}$, 待查询的多度量距离函数 $\overline{d}(\cdot,\cdot)$;
输出: 多度量 ϵ-近邻查询结果集 ans.

1. $que \leftarrow \{\overline{q}\}$ /*初始化优先队列 que */
2. $ans \leftarrow \{\overline{q}\}$ /*初始化结果集 ans */
3. **while** $que \neq \varnothing$ **do**
4. $u \leftarrow que.top()$ /*找到第 1 个元素*/
5. $que.pop()$
6. **for each** v, v 是 u 的 k-近邻 **do**
7. **if** $\overline{d}(v,\overline{q}) < \epsilon$ **then**
8. $ans \leftarrow ans \cup \{v\}$
9. $que \leftarrow que \cup \{v\}$
10. **return** ans

算法 3 的输入为聚合多度量图索引 $\overline{G}$, 近邻距离限制参数 ϵ, 查询数据对象 $\overline{q}$, 以及待查询的多度量距离函数 $\overline{d}(\cdot,\cdot)$. 算法首先初始化优先队列 que 为待查询的数据对象 $\overline{q}$, 初始化结果集 ans 为查询对象 $\overline{q}$. 接着, 其执行 while 循环, 直到队列 que 为空. 在每一次循环迭代中, 基于最优策略优先的思想, 找到当前的队首元素 u (第 4, 5 行). 接着, 找到图中所有与 u 存在近邻关系的数据对象 v (第 6 行). 若 v 满足 $\overline{d}(v,\overline{q}) < \epsilon$ (即 v 是多度量 ϵ-近邻查询结果), 则将 v 加入结果集 ans, 并且将其加入优先队列 que (第 8, 9 行). 最终, 当队列 que 为空时, 循环结束, 并且算法返回多度量 ϵ-近邻查询结果集 ans.

注意到, 在算法 2 中第 11 行与算法 3 中第 7 行都是直接计算距离从而判断是否是 k-近邻/ ϵ-近邻. 然而, 在实际计算中, 该距离计算过程可以基于三角不等式进行优化. 例如, 给定查询数据对象 $\overline{q}$, 一个已经访问过的数据对象 u 以及 u 的 k-近邻 v. 由于 $\overline{d}(u,\overline{q})$ 在先前访问数据对象 u 的时候已经计算, $\overline{d}(u,v)$ (即, 数据对象 u 和 v 的近邻关

系) 在构建聚合多度量图时已经进行计算, 因此可以依据多度量距离函数 $\overline{d}(\cdot,\cdot)$ 的三角不等特性对 $\overline{d}(v,\overline{q})$ 进行估算, 得到 $|\overline{d}(v,u)-\overline{d}(u,\overline{q})|\leqslant\overline{d}(v,\overline{q})\leqslant\overline{d}(v,u)+\overline{d}(u,\overline{q})$. 因此, 通过该不等式可以避免部分不必要的距离计算次数, 从而提升算法效率.

尽管基于算法 2 与算法 3 中描述的聚合多度量图构建方法与多度量 ϵ-近邻查询方法可以大幅提升算法 1 的效率, 然而算法 3 中多度量查询的结果依然有较大的可能性不是真实结果. 图 6 描绘了一个基于图 5 中的多度量空间数据进行 ϵ-近邻查询的示例. 其中, 查询对象为 $\overline{p}_4$, $\epsilon=3$. 注意到, 根据算法 3 中的过程, 首先以查询数据对象为基础对外扩张, 得到第 1 个近邻 $\overline{p}_2$ 并且将其作为结果加入结果集合中. 接着, 以 $\overline{p}_2$ 为基础向外寻找近邻关系找到数据对象 $\overline{p}_1$, 但是由于 $\overline{p}_1$ 与查询对象 $\overline{p}_4$ 的距离超过查询半径, 因此不将 $\overline{p}_1$ 加入队列中. 至此, 队列为空, 返回所有查询结果 $\overline{p}_2$ 和 $\overline{p}_4$. 然而, 注意到返回的结果中遗漏了数据对象 $\overline{p}_3$ ($\overline{d}(\overline{p}_3,\overline{p}_4)=2.75\leqslant\epsilon$, 因此 $\overline{p}_3$ 也是该查询的结果). 因此, 算法 3 中给出的多度量空间 ϵ-近邻查询方法的精确性仍然存疑. 对此, 其原因可以归结为近邻关系的不可传递性, 即近邻的近邻不一定也是近邻. 考虑到本文在构建聚合多度量图中, 采用差分化距离的策略进行图的构建 (即, 先插入的近邻关系可能是不相似的数据对象因此距离较长, 而后插入的近邻关系往往是精确的近邻关系因此距离较短). 类似地,在查询中也可以采用类似的方法以提高查询精度. 具体而言, 不同于现有方法通过从候选最近邻结果展开局部搜索的查询策略, 本文采用全局搜索并通过聚合多度量图中的差分化距离关系辅助搜索, 以实现更精确的多度量 ϵ-近邻查询 (例如, 在进行对 $\overline{p}_4$ 的查询时, 从 $\overline{p}_1$ 开始搜索其近邻, 这样可以直接找到正确的结果 $\overline{p}_2$ 和 $\overline{p}_3$).

多度量空间 $\overline{S}$	度量空间 S_1	度量空间 S_2
$\overline{p}_1$	p_1^1=(2, 2)	p_1^2=5
$\overline{p}_2$	p_2^1=(1, 0)	p_2^2=10.0
$\overline{p}_3$	p_3^1=(4, 1)	p_3^2=3.5
$\overline{p}_4$	p_4^1=(4, 0)	p_4^2=8.0

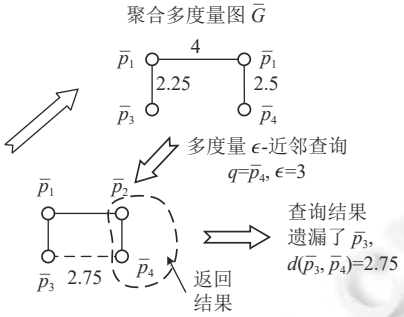


图 6 多度量 ϵ-近邻查询示例 (查询对象为 $\overline{p}_4$)

具体而言, 在算法 3 中, 其核心思想为基于查询数据对象向外扩张. 虽然该方法效率较高, 但是由于近邻关系不具有传递性, 因此是局部搜索结果, 精度较差. 而本文提出的聚合多度量图中存在长短边关系, 可以考虑从全局角度切入, 进行全局查找从而提升查询效率. 尽管如此, 通过全局查询方法以提升查询精度存在效率瓶颈, 即需要从图中一个与查询对象不相关的位置展开搜索, 会造成高昂的图遍历开销. 尽管每次遍历时都会依据最短的距离展开遍历搜索, 但是因为图遍历导致的节点扩张开销依然会严重影响多度量数据聚类的效率. 此外, 如何判断是否扩张亦是一个难题. 例如, 在进行 k-近邻/ϵ-近邻查询时都可以依据距离限制进行剪枝, 筛除不必要的数据对象. 然而在进行全局 ϵ-近邻查询时, 如何设定距离限制以进行剪枝是不可获知的. 距离限制过大将导致大量的节点被访问, 性能较差; 距离限制过小将导致在初始扩张阶段便因为当前所有数据对象到查询对象的距离均超过距离限制而终止图遍历.

鉴于此, 本文提出图构建与图查询同时进行的策略, 以解决因全局搜索带来的性能瓶颈以及距离限制难以给定的难题. 具体而言, 在进行全局 k-近邻查询以构建聚合多度量图时, 同时进行 ϵ-近邻查询. 当访问的数据对象是 k-近邻/ϵ-近邻二者其一时, 即将该数据对象加入队列中进行扩张. 因此, 通过该方法进行聚合多度量图的建立时, 每个数据对象的 ϵ-近邻也在 k-近邻的过程中被记录, 并且该方法同时支持后续数据聚类的操作.

算法 4 描述了基于多度量图 (AMG) 的多度量空间 DBSCAN 算法, 其输入为待查询的多度量空间 $(\overline{S},\overline{d})$, 近邻

参数 k, 近邻距离限制参数 ϵ 以及近邻数量限制参数 $minpts$. 算法首先初始化多度量图索引 $\overline{G}$ 为多度量空间中的第 1 个数据对象 $\overline{p}_{ini}$. 接着, 对于多度量空间中剩下的每个数据对象 $\overline{p}$ (第 2 行), 算法基于现有的多度量图索引 $\overline{G}$ 找到其多度量 ϵ-近邻与 k-近邻, 而后根据近邻数量限制 $minpts$ 判断 $\overline{p}$ 是否为核数据对象 (第 3–6 行). 接着, 对于 $\overline{p}$ 的所有 ϵ-近邻, 再次判断这些数据对象是否是核数据对象 (第 7–10 行). 注意到, 即使有些数据对象在先前的结果中并不满足基于 $minpts$ 的核数据对象条件, 但是其和新加入的数据对象 $\overline{p}$ 存在 ϵ-近邻关系, 因此有可能成为新的核数据对象, 所以需要对这些先前的数据对象再次进行判定. 接着, 算法对所有的核数据对象进行基于密度的关联 (第 11–18 行). 最终, 算法返回所有找到的不重复聚类的集合 C (第 19–22 行).

算法 4. 基于聚合多度量图 (AMG) 的多度量空间 DBSCAN 算法.

输入: 待查询的多度量空间 $(\overline{S},\overline{d})$, 近邻参数 k, 近邻距离限制参数 ϵ, 近邻数量限制参数 $minpts$;
输出: 多度量空间内所有基于密度的聚类.

1. $\overline{G} \leftarrow \{\overline{p}_{\text{ini}}\}$
2. **for each** $\overline{p} \in \overline{S}$, 且 $\overline{p} \neq \overline{p}_{\text{ini}}$ do
3. $N_\epsilon(\overline{p}), N_k(\overline{p}) \leftarrow find(\overline{p}, \overline{G}, \overline{p}_{\text{ini}}, k, \epsilon)$ /*计算 $\overline{p}$ 的 $N_\epsilon(\overline{p})$ 或者所有 k-近邻 (记为 $N_k(\overline{p})$)*/
4. 将 $\overline{p}$ 与其所有 k-近邻数据对象 $\overline{p}' \in N_k(\overline{p})$ 在聚合多度量图索引 $\overline{G}$ 中建立近邻关系
5. **if** $N_\epsilon(\overline{p}) \geqslant minpts$ **then** /*核数据对象*/
6. 标记 $\overline{p}$ 为核数据对象
7. **for each** $\overline{p}' \in N_\epsilon(\overline{p})$ **do**
8. $N_\epsilon(\overline{p}') \leftarrow N_\epsilon(\overline{p}') \cup \{\overline{p}\}$
9. **if** $|N_\epsilon(\overline{p}')| \geqslant minpts$ **then** /*该数据对象是核数据对象*/
10. 标记 $\overline{p}'$ 为核数据对象
11. **for each** 核数据对象 $\overline{c} \in \overline{S}$ do
12. $C_{\overline{c}} \leftarrow |\overline{c}|$
13. **for each** $\overline{p} \in N_\epsilon(\overline{c})$ **do**
14. **if** $|N_\epsilon(\overline{p})| \geqslant minpts$ **then**
15. $C_{\overline{p}} \leftarrow \overline{p}$ 对应的聚类
16. $C_{\overline{c}} \leftarrow C_{\overline{c}} \cup C_{\overline{p}}$
17. **else**
18. $C_{\overline{c}} \leftarrow C_{\overline{c}} \cup \overline{p}$
19. $C \leftarrow \varnothing$
20. **for each** 核数据对象 $\overline{c} \in \overline{S}$ **do**
21. $C \leftarrow C \cup C_{\overline{c}}$
22. **return** 所有找到的多度量空间聚类 C

算法 5 描述了基于聚合多度量图 (AMG) 的多度量 ϵ-近邻与 k-近邻的查询算法, 其输入为查询数据对象 $\overline{q}$, 聚合多度量图索引 $\overline{G}$, $\overline{G}$ 中的第 1 个数据对象 $\overline{p}_{\text{ini}}$, 近邻参数 k 以及近邻距离限制参数 ϵ. 算法首先初始化优先队列 que 为多度量图索引 $\overline{G}$ 为第 1 个多度量数据对象 $\overline{p}_{\text{ini}}$, $\overline{q}$ 的局部 k-近邻数据对象集合为空集, 将 $\overline{q}$ 的多度量 ϵ-近邻数据对象集合 N_ϵ 初始化为空集以及初始化局部距离最值 dis_k 为无穷大 (第 1–4 行). 接着, 算法执行 while 循环, 直到队列 que 为空. 在每一次循环迭代中, 对于当前的队首元素 u (距离查询对象 $\overline{q}$ 最近的数据对象), 算法判断当前数据对象是否是 $\overline{q}$ 的多度量 ϵ-近邻或者 k-近邻, 并且进行相应的更新 (第 6–14 行). 即, 若当前对象是多度量 ϵ-近邻, 则加入结果集 N_ϵ (第 9 行); 若当前对象是多度量 k-近邻, 则更新 k-近邻结果集 N_k 并且更新队列 que (第 10–14

行). 最终, 当队列 que 为空时, 循环结束, 并且算法返回查询数据对象 $\overline{q}$ 的多度量 ϵ-近邻结果 N_ϵ 与多度量 k-近邻结果 N_k (第 15 行).

算法 5. 多度量 ϵ-近邻与 k-近邻的查询算法.

输入: 查询数据对象 $\overline{q}$, 聚合多度量图索引 $\overline{G}$, $\overline{G}$ 中的第 1 个数据对象 $\overline{p}_{ini}$, 近邻参数 k, 近邻距离限制参数 ϵ;
输出: 查询数据对象 $\overline{q}$ 的多度量 ϵ-近邻 $N_\epsilon(\overline{q})$ 与 k-近邻 $N_k(\overline{q})$.

1. $que \leftarrow \{\overline{p}_{\text{ini}}\}$ /*初始化优先队列 que */
2. $N_k \leftarrow \varnothing$ /*初始化结果集 N_k 用于存放 $\overline{q}$ 的所有 k-近邻数据对象*/
3. $N_\epsilon \leftarrow \varnothing$ /*初始化结果集 N_ϵ 用于存放 $\overline{q}$ 的所有 ϵ-近邻数据对象*/
4. $dis_k \leftarrow \infty$ /*当前访问过的数据对象中距离 $\overline{q}$ 第 k 远数据对象到 $\overline{q}$ 的距离, 即 N_k 中距离 $\overline{q}$ 最远数据对象的距离*/
5. **while** $que \neq \varnothing$ **do**
6. $u \leftarrow que.top()$ /*找到第 1 个元素*/
7. $que.pop()$
8. **if** $\overline{d}(u,\overline{q}) < dis_k$ 或 $\overline{d}(u,\overline{q}) < \epsilon$ **then**
9. **if** $\overline{d}(u,\overline{q}) < \epsilon$ **then** $N_\epsilon \leftarrow N_\epsilon \cup \{u\}$
10. **if** $\overline{d}(u,\overline{q}) < dis_k$ **then**
11. 用 u 更新 N_k
12. 用 $\overline{d}(u,\overline{q})$ 更新 dis_k
13. **for each** v, v 是 u 的 k-近邻 **do**
14. $que \leftarrow que \cup \{v\}$
15. **return** 查询数据对象 $\overline{q}$ 的多度量 ϵ-近邻结果 N_ϵ 与多度量 k-近邻结果 N_k

基于本文提出的算法 4 进行多度量空间 DBSCAN 聚类的时间复杂度分为两部分, 第 1 部分为构建多度量图的时间, 即算法 5 中多度量 ϵ-近邻与 k-近邻的查询复杂度. 根据近邻图综述[10]中对基于数据插入的近邻图构建方法与查询方法时间复杂度分析[37], 本文中提出的查询方法与构建方法时间分别是 $O(\log^2(k\cdot|\overline{S}|))$ 与 $O(|\overline{S}|\log^2(k\cdot|\overline{S}|))$. 第 2 部分是将所有的核数据对象进行连接形成多度量空间数据聚类. 因此, 仅需遍历聚合多度量图中的所有近邻对象 (即算法 4 第 13 行) 而后进行聚类合并即可. 该步骤时间复杂度与所有数据对象的多度量 ϵ-近邻数量一致, 因此不超过查询复杂度. 综上, 本文提出的多度量空间聚类算法时间复杂度为 $O(|\overline{S}|\log^2(k\cdot|\overline{S}|))$. 此外, 由于聚合多度量图仅保存数据对象与其中的相似性关系 (即图中的边), 因此算法的空间复杂度为 $O(k\cdot|\overline{S}|)$.

4 实验分析

本节在 4 个数据集上从距离计算次数、运行时间以及查询结果精确度 3 个方面对所提出的基于密度的多度量空间聚类算法展开实验评估. 第 4.1 节介绍了本文的实验数据、基线对比方法、实验参数与评价指标. 第 4.2 节对算法构建参数选择展开讨论. 第 4.3 节对算法的查询性能结果展开分析. 第 4.4 节对所提出算法的可扩展性进行分析. 第 4.5 节对所有算法的聚类效果进行分析. 第 4.6 节给出了一个在真实场景下进行多度量空间数据聚类的案例分析.

4.1 实验数据

● 实验数据集. 本文使用 3 个公开的真实数据集 (食物数据集 Food[38]、空气质量数据集 Air[39]以及租房数据集 Rental[40]) 与一个合成数据集进行实验测试. 其中, 食物数据集包括多种食物的添加剂、诸多营养成分、图片以及对于食物的描述等 9 种信息; 空气质量数据集中包括了在印度多个城市的多个位置随机测出的 NO、NO_2、甲苯等 9 种空气污染物成分在空气中的含量; 租房数据集中罗列了部分位于美国纽约的房租信息, 包括每间房子的

房型、位置、公开时间、租房价格、租客的描述以及房屋的照片信息. 此外, 本文还基于图片数据集[41]、洛杉矶地理信息[42]、文本数据集[43]以及随机生成的特征向量合成了一个由 50 种不同类型数据组成的合成数据集. 表 1 给出了所使用数据集的统计信息, 以及在本实验中每个数据集中每种数据类型对应的距离度量方式. 其中, 对于食物数据集中的食物描述信息以及租房数据集中的租客评价, 我们采用词向量余弦距离度量[44]的方法对其进行相似度计算. 此外, 本文还基于图片数据[41]、洛杉矶地理信息[42]、文本数据集[43]以及随机生成的特征向量合成了一个由 50 种不同类型数据组成的合成数据集. 表 1 给出了所使用数据集的统计信息, 以及在本实验中每个数据集中每种数据类型对应的距离度量方式. 其中, 对于食物数据集中的食物描述信息以及租房数据集中的租客评价, 我们采用词向量余弦距离度量[44]的方法对其进行相似度计算.

表 1 数据集汇总

名称	数据规模	数据类型数	距离度量方式
食物数据集Food[38]	38 759	9	L_1距离: 食物添加剂、营养成分以及图片 编辑距离: 食物种类信息 词向量余弦距离[44]: 食物描述信息
空气质量数据集Air[39]	100 000	9	L_1距离: 所有污染物含量
租房数据集Rental[40]	113 176	5	L_1距离: 价格、公布日期以及图片 L_2距离: 房型以及地理位置 词向量余弦距离[44]: 租客评价
合成数据集Sync	50 000	50	L_1距离: 数值特征以及图片 L_2距离: 地理位置 编辑距离: 文本信息

● 基线对比方法. 为了充分评估所提出算法的有效性, 实验选择了 2 种不同类型的多度量空间聚类方法作为基准, 分别是: 基于现有多度量空间融合型索引 RR*-tree[21]与基于多度量空间分离型索引 DESIRE[22]实现的多度量空间 DBSCAN 方法. 此外, 在实验中本文报道直接通过聚合多度量图进行图构建而后进行数据聚类的方法 AMG (即, 在算法 1 框架中使用算法 2 作为聚合多度量图构建方法并且通过算法 3 实现多度量空间 ϵ-近邻查询), 与改进的聚合多度量图 DBSCAN 方法 AMG* (即算法 4) 的查询结果. 实验硬件平台为: Intel(R) Core(TM) i7-7700 CPU@3.6 GHz, 32 GB 内存, 系统平台为 Windows 10.

● 实验设置与评价指标. 实验采用不同的实验设置与评价指标对所提出的方法进行全面评估. 通过变化聚合多度量图索引方法中构建图采用的近邻参数 k、多度量空间 DBSCAN 中选用的查询参数 ϵ 与 *minpts*、待查询的度量空间数量、查询权重 w_r (例如, 给定两个查询空间对应的权重 w_1 与 w_2 与查询权重 w_r, 则 $w_1/w_2 = w\_r$) 以及数据集的规模. 实验报告: (1) 距离计算次数, 即对数据对象对基于给定的任意单模态度量方式进行一次相似性计算的次数 (一次多度量计算是多个单模态度量计算的综合); (2) 运行时间, 即通过算法进行数据聚类的 CPU 运行时间; (3) 精度, 即所有找到的数据聚类相比所有的数据聚类的个数. 注意到, 由于基于密度的数据聚类由两部分组成 (即数据对象与其内部相似性关系), 并且通过其中的相似性关系可以完全还原出所有的聚类. 因此, 记算法找到的满足 ϵ-近邻关系的相似性关系数量为 N_1, 真实存在的满足 ϵ-近邻关系的相似性关系数量为 N_2, 则算法的精度为 $N_1/N_2 \times 100\%$. 其中真实存在的 ϵ-近邻关系 N_2 可以通过精确算法 (如 DESIRE 算法) 得到. 并且由于任何由本文提出的近似算法得到相似性关系均满足实际的 ϵ-近邻关系, 因此 $N_1 \leqslant N_2$, 即精度不会超过 100%. 此外, 本文同时结合类内距离平方均值 (WSS)、类间距离平方均值 (BSS) 以及通用的聚类分析研究指标[45]轮廓系数 (SC)[46]对本文算法得到的聚类效果进行分析. 表 2 给出了所使用实验参数的统计信息, 以及其默认值大小. 其中, 范围参数 ϵ 的取值为在给定查询的多度量空间中相似性距离平均值的 1‰, 2‰, 4‰, 8‰, 16‰, 32‰, 而 *minpts* 取值为多度量空间数据对象的 ϵ-近邻数量平均值的 0.2 倍, 0.4 倍, 0.6 倍, 0.8 倍以及 1 倍. 例如, 给定待查询的多度量空间中距离平均值为 10 000, 取 ϵ=8‰, 则真实的 ϵ=80. 若给定待查询的多度量空间中数据对象平均的 ϵ-近邻数为 40, 取 *minpts*=0.6, 则真实的 *minpts*=240.

表 2 实验参数设置

实验参数	数值	默认值
图构建近邻参数k	10, 20, 30, 40, 50	30
范围参数 ϵ	1‰, 2‰, 4‰, 8‰, 16‰, 32‰	8‰
minpts	0.2, 0.4, 0.6, 0.8, 1	0.6
待查询的度量空间数	1, 2, 3, 4	2
查询权重w_r	0.1, 0.5, 1, 5, 10	1
数据集的规模	20%, 40%, 60%, 80%, 100%	100%

4.2 算法参数设置

本文所提出的聚合多度量图索引算法在进行近邻图构建时采用了参数 k. 其与图的稠密度相关, 因此影响着多度量空间 DBSCAN 的效率与精度. 如图 7 所示, 随着参数 k 的增长, 运行时间与距离计算次数逐渐增加, 而查询结果也愈发精确. 这是因为每个数据对象都要与更多的数据对象建立相似联系, 因此需要访问更多的数据对象, 造成运行时间与距离计算次数增加. 此外, 随着更多的数据对象被访问, 聚合多度量图的搜索范围更大, 因此找到的多度量空间数据聚类结果也更加精确. 因此, 为保证在进行多度量空间数据聚类时, 同时具有较高的查询效率和精确的数据聚类效果, 本文采用 k=30 作为默认的查询参数.

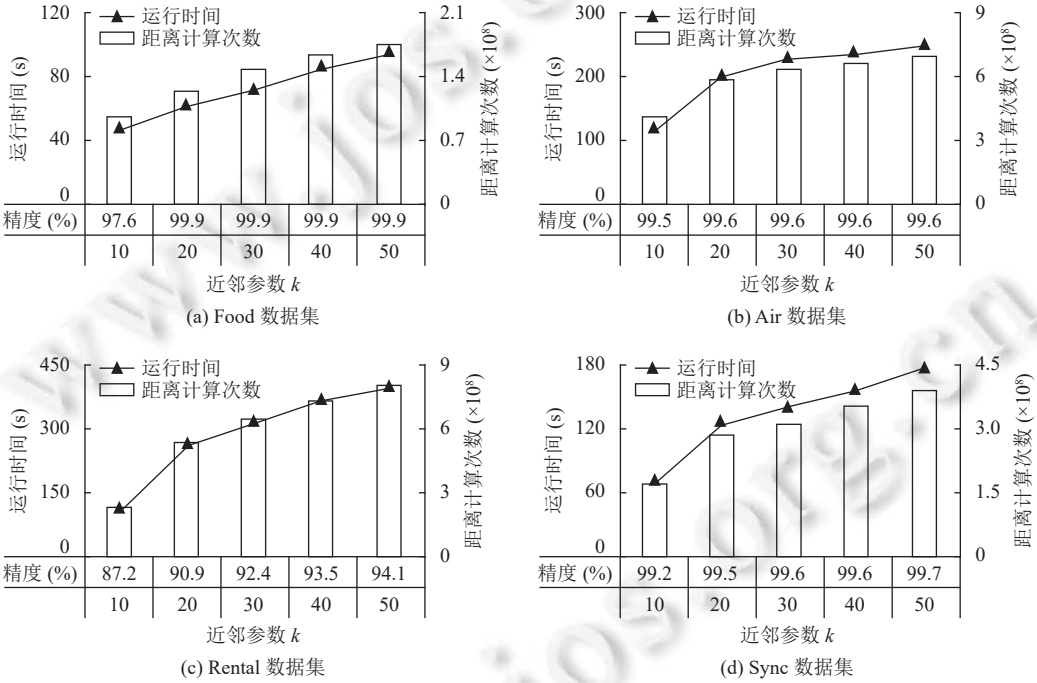


图 7 图构建近邻参数 k 的变化

4.3 算法聚类性能实验结果与分析

● 查询半径 ϵ 的变化. 本节在所有数据集上变化查询半径对距离计算次数、算法运行时间以及查询结果精度的影响. 如图 8 所示, 本文所提出的 AMG 与 AMG*算法在所有数据集上查询性能超越现有的多度量空间数据索引 DESIRE 与 RR*-tree, 并且 AMG*算法超越现有方法一个数量级以上. 这是因为不同于树结构需要进行层次遍历以找到所需要的结果, 图结构可以更快速地定位查询对象所在区域. 因此, 在进行多度量空间 DBSCAN 时可以更快地找到每个数据对象的所有 ϵ-近邻. 并且随着查询范围的增加, 所提出的图索引方法 AMG 与 AMG*没有显著的线性增加关系, 这是因为所提出算法需要进行构图操作. 在进行图构建的时候, 一部分数据对象的 ϵ-近邻对象

也被遍历, 因此其主要的时间开销来源于图构建操作, 且随着距离限制 ϵ 逐渐增大 (例如在 Food 数据集中达到 32‰时, 时间开销也呈现增长趋势). 同时, 注意到在距离计算次数上, Food 数据集与 Sync 数据集中 AMG*是所有算法中距离计算次数最少的, 而在 Air 以及 Rental 数据上 RR*-tree 在部分条件下具有较少的距离计算次数. 这是因为 RR*-tree 利用支枢点进行剪枝, 从而避免不必要的距离计算次数. 尽管如此, AMG*的性能依然远超过 RR*-tree, 这说明距离计算次数虽然和算法运行时间有一定关系, 但不是决定性因素. 此外, 在所有数据集上, AMG*算法的精确度均远远超过 AMG 算法, 这验证了本文提出的关于图构建与数据聚类操作不同步将导致查询结果不精确的结论, 并且证实了本文提出的 AMG*算法的有效性.

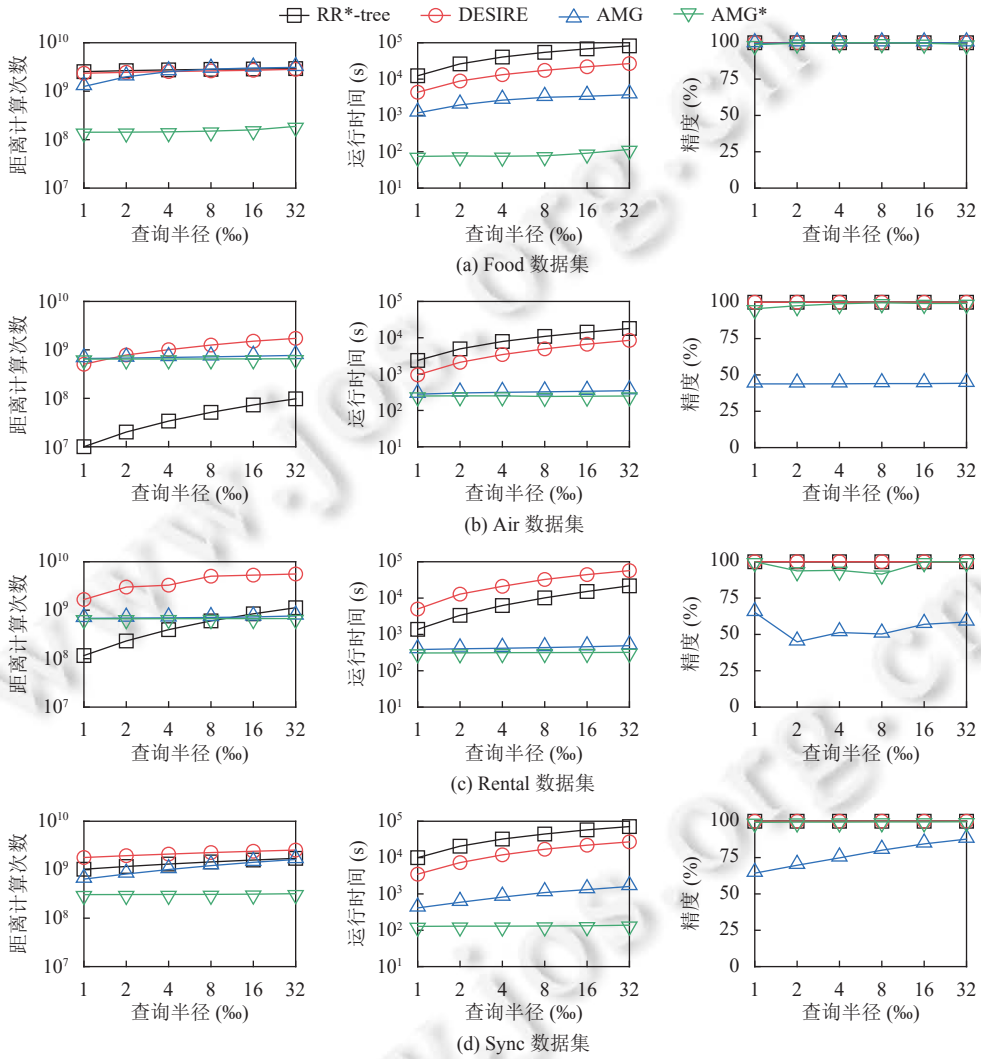
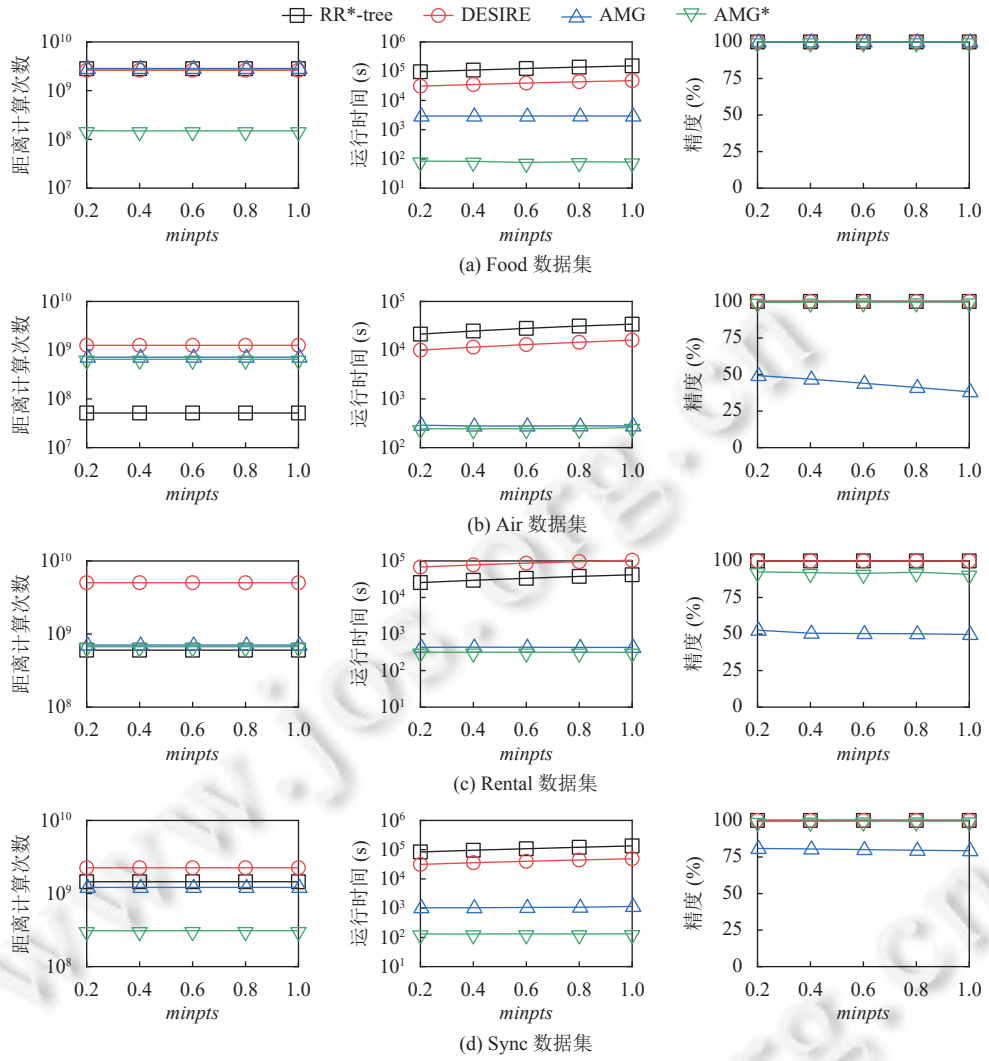


图 8 查询半径 ϵ 的变化

● 查询参数 *minpts* 的变化. 本文接着测试随着数量限制参数 *minpts* 的改变, 各算法的性能波动. 注意到, 该参数的变化并不影响多度量 ϵ-近邻查询的性能, 而只会对聚类合并的过程产生影响. 因此, 如图 9 所示, 随着 *minpts* 的增加, 所有算法距离计算次数不变, 而运行时间有轻微的上升趋势. 不仅如此, 所提出的聚合图索引方法 AMG 的精度都有所下降, 但是 AMG*的精度依然保持平稳. 这是因为不同于 AMG 使用的局部搜索策略, AMG*采用全局搜索进行多度量 ϵ-近邻查询, 因此具有较高的精度且覆盖更广的查询范围, 从而不受查询参数影响.



(a) Food 数据集

(b) Air 数据集

(c) Rental 数据集

(d) Sync 数据集

图 9 *minpts* 的变化

4.4 算法可扩展性结果与分析

● 待查询度量空间数量的变化. 为了探究算法的可扩展性, 本文首先变化查询空间的数量. 如图 10 所示, 随着度量空间数的增加, 查询时间和距离计算次数逐渐增加. 并且注意到当度量空间=1 时 (即, 单度量空间) 下, DESIRE 方法在 Air 数据集上的运行时间超过了本文提出的 AMG 与 AMG*方法. 这是因为在该数据集下, 数据存在大量冗余. 因此利用传统的索引与其剪枝策略可以更快速地进行数据聚类. 然而, 在多度量空间中 (即, 度量空间数>1 时), 所提出的方法性能远远超过现有方法. 此外, 随着度量空间数的增加, AMG 与 AMG*的精度呈现波动趋势, 这是因为不同度量空间的特性不同, 而应对高冗余数据时 (如 Air 数据集), 近邻图由于仅保存最为相似的数据对象相似性关系, 因此存在精度缺陷. 尽管如此, AMG*方法依然保持着 85% 以上的精度.

● 查询权重 w_r 的变化. 接着, 本文变化查询权重 w_r 以测试应对用户个性化需求时的查询性能. 如图 11 所示, 随着查询权重的变化, 本文所提出的 AMG 方法与 AMG*方法均具有卓越的查询性能, 而 AMG*依然超越所有现有方法 1–2 个数量级. 此外, 注意到 AMG 方法在部分条件下精度有所波动 (例如在 Rental 数据集中 w_r=10 时). 然而, 尽管 AMG*方法的精度有所波动, 其精度依然较高 (均≥85%). 此结果再次验证了本文所提出的全局搜索方法相较于局部搜索具有更高的精度, 并且表明所提出的差分化相似性特征表征方法的有效性.

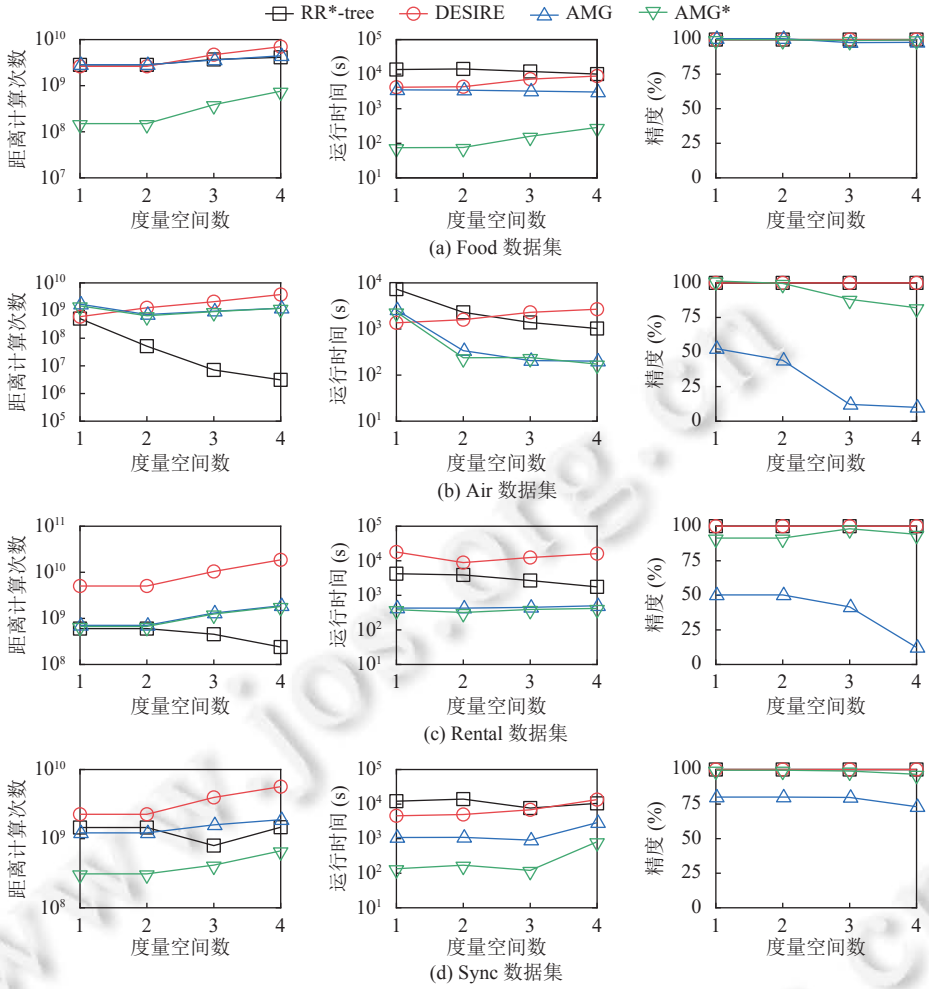


图 10 待查询度量空间数量的变化

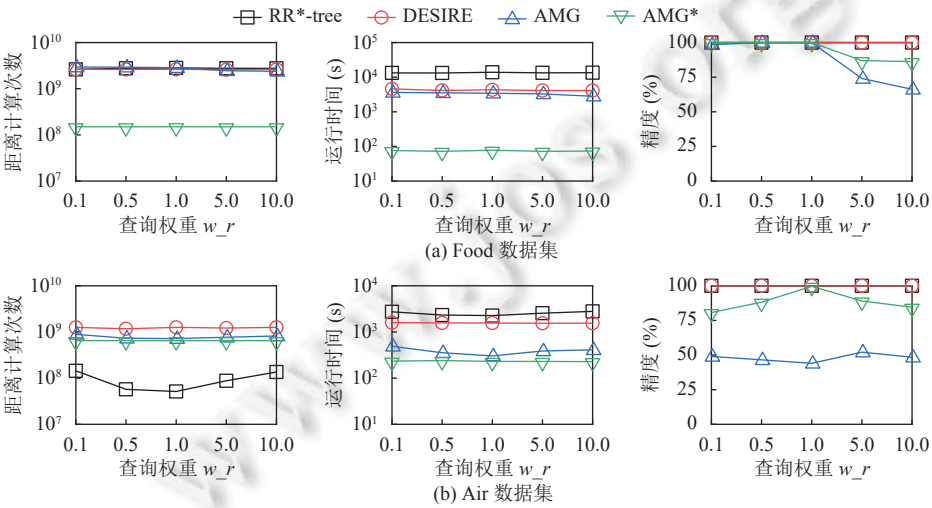


图 11 查询权重 w_r 的变化

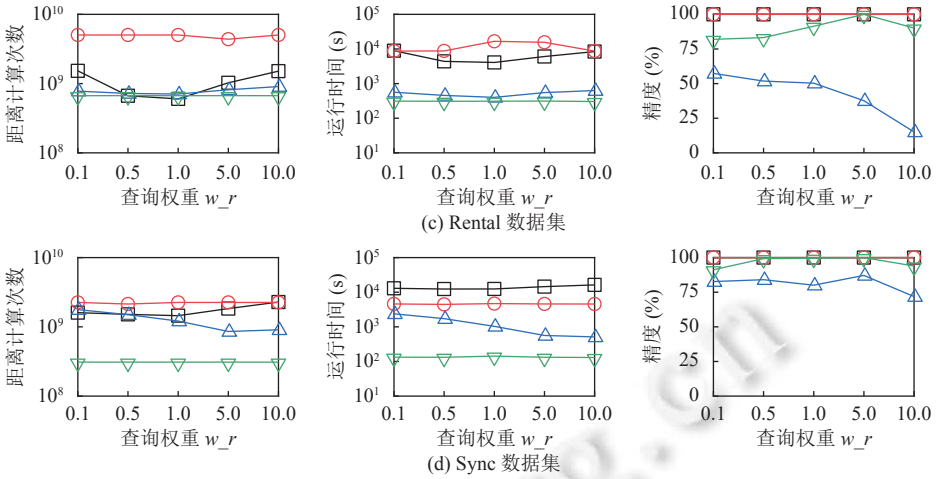


图 11　查询权重 w_r 的变化 (续)

● 数据规模的变化. 最后, 为了评估本文提出方法的可扩展性, 本文变化数据集规模 (从 20% 变化至 100%, 如图 12 所示). 实验结果表明, 随着数据集规模的上升, 本文提出的 AMG*方法时间也随之增长, 并且均超越基于现有多度量空间数据索引的聚类方法. 不仅如此, 在部分数据集中, 随着数据规模的上升, 算法的精度也有所上升 (例如 Air 数据集与 Rental 数据集). 这是因为在数据分布特征相似的数据集中, 随着数据规模的增加, 近邻图也愈发稠密, 因此能够更好地刻画数据间的相似性特征. 因此, 随着数据规模的增加, AMG*能够更精确地找到每个数据对象的多度量 ϵ-近邻, 从而提升结果精度.

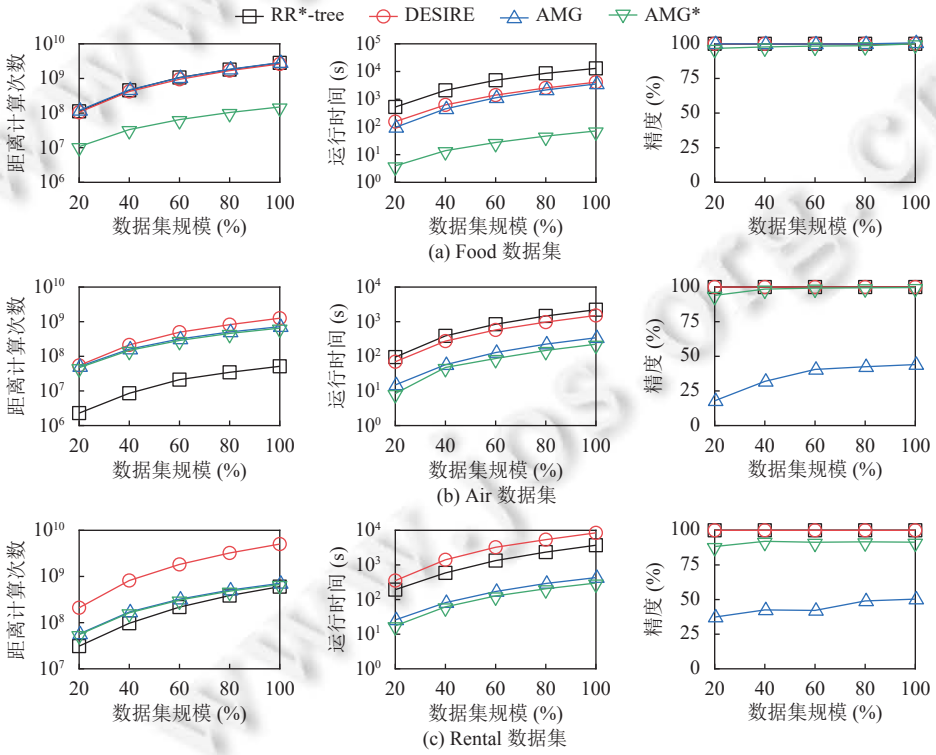
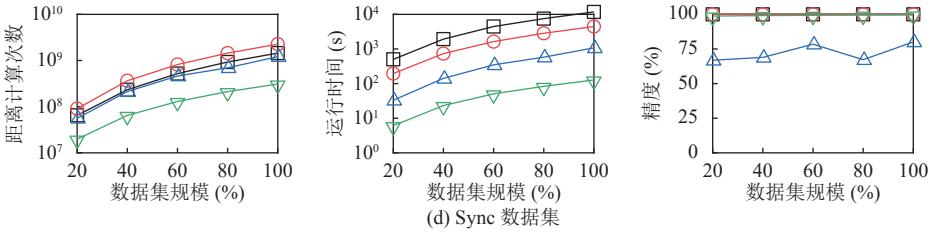


图 12　数据集规模的变化



(d) Sync 数据集

图 12 数据集规模的变化 (续)

4.5 数据聚类效果分析

本节采用聚类结果中的节点数、边数、变数与节点数的比值、类内距离平方均值 (WSS)、类间距离平方均值 (BSS)、类间距离平方均值与类内距离平方均值的比值以及通用的聚类分析研究指标[45]轮廓系数 (SC)[46]对所有算法在 4 个数据集上按照表 2 所示的默认参数进行 DBSCAN 得到的聚类进行分析. 从表 3 所示的结果中可以看出: 采用本文所提出的 AMG*算法得到的聚类节点数与边数均与精确方法 (DESIRE 与 RR*-tree) 得到的结果极为接近, 验证了本文提出算法的有效性. 此外, 所有算法得到的聚类结果均具有较高的 BSS 与 WSS 比值, 且轮廓系数均大于 0. 这些实验结果说明通过对多度量空间进行 DBSCAN 实现了高质量的数据聚类. 值得注意的是, 尽管 AMG 方法所获得数据聚类精度较低 (例如, 在 Rental 数据集上相比于精确算法得到的边数较少), 其所得到的聚类依然具有较高的 BSS 与 WSS 比值, 以及较高的 SC 值. 这是因为尽管其对数据聚类得到的边数较少, 其计算得到的节点数相较于精确算法得到的节点数没有较大的偏差. 因此, 通过 AMG 方法依然可以得到较好的数据聚类效果.

表 3 聚类效果比较

数据集	算法	点数	边数	边数/点数	WSS	BSS	BSS/WSS	SC[46]
Food	RR*-tree	31 613	6 242 348	197.5	0.919 3	1.080 6	1.175 4	0.110 3
	DESIRE	31 613	6 242 348	197.5	0.919 3	1.080 6	1.175 4	0.110 5
	AMG	31 604	6 212 381	196.6	0.919 1	1.080 4	1.175 5	0.110 3
	AMG*	31 539	6 169 003	195.6	0.926 6	1.200 5	1.295 6	0.141 2
Air	RR*-tree	55 929	9 163 569	163.8	0.000 7	0.001 3	1.715 1	0.335 6
	DESIRE	55 929	9 163 569	163.8	0.000 7	0.001 3	1.715 1	0.329 2
	AMG	51 882	3 968 554	76.5	0.000 6	0.001 5	2.362 1	0.404 0
	AMG*	55 243	8 976 406	162.5	0.000 7	0.001 5	2.039 2	0.374 8
Rental	RR*-tree	26 397	15 726 364	595.8	0.202 0	1.942 8	9.618 3	0.753 5
	DESIRE	26 397	15 726 364	595.8	0.202 0	1.942 8	9.618 3	0.746 6
	AMG	24 664	8 559 849	347.1	0.202 8	1.953 3	9.632 3	0.744 8
	AMG*	25 914	15 561 504	600.5	0.204 1	1.940 2	9.508 3	0.736 2
Sync	RR*-tree	8 525	840 266	98.6	0.543 0	1.465 9	2.699 5	0.428 5
	DESIRE	8 525	840 266	98.6	0.543 0	1.465 9	2.699 5	0.428 5
	AMG	7 750	534 680	69.0	0.536 7	1.158 0	2.157 6	0.367 3
	AMG*	8 419	833 308	99.0	0.586 7	1.493 2	2.545 3	0.398 6

4.6 真实场景数据聚类案例分析

本节介绍了一个在房屋租赁场景下利用本文提出的多度量空间数据算法进行数据聚类的案例 (如图 13 所示). 在包括了纽约租房信息的 Rental 数据集下, 本文同时对价格信息、房型信息、房屋发布时间以及房屋环境照片进行数据聚类. 其中, 所有信息均采用 L_1 距离进行距离度量, 并且每个度量空间下的权重均设置为 1 (即, 每个独立信息都一样重要). 图中左边展现了一个围绕市区中央的房屋聚类, 右边是一个围绕郊区的房屋聚类. 在每个聚类中, 这些房型都与聚类中的某些房屋在给定的查询度量空间上具有一定的相似性, 从而满足不同的个性化需求. 其中, 图中左半部分中通过数据聚类得到的房屋具有以下特征: 均价在 3 000 美元左右, 房屋发布时间在 2016 年 5 月, 在均具有较为鲜明的房屋装修风格的同时都处于市中心的位置. 相对的, 右半部分得到的房屋聚类则坐落

于城郊, 其租赁价格普遍较低, 房屋发布时间较新且房屋装修风格偏冷色调. 由此可见, 多度量空间数据聚类可以根据不同的需求和特征将房屋聚类成不同的群组, 以便满足个性化的需求. 因此, 其具有广阔的实际应用场景.

图 13 真实场景数据聚类案例

5 结论与展望

本文提出了一种基于密度的多度量空间数据聚类方法, 从而支持高效的多模态数据聚类. 首先, 本文设计了一种基于聚合多度量图的数据表征方法, 从而刻画不同模态数据间的相似性关系. 其次, 本文结合差分化的相似性关系提出了高效的聚合图构建方法, 从而优化图结构中的数据分布, 以提升图的构建性能以及多度量空间数据聚类的准确性. 此外, 本文提出了结合最优策略优先的 ϵ -近邻查询方法, 提升了多度量空间数据聚类的性能. 最后, 在公开数据集与合成数据集上进行了充分实验, 实验结果表明所提出的基于聚合多度量图的多度量空间数据聚类方法 AMG*大幅提升了多度量空间数据聚类的效率, 并且具有较高的精度与良好的可扩展性.

未来的工作包括 3 个方面: (1) 基于分布式的超大规模多度量空间数据聚类, 以支持更为庞大的数据规模; (2) 基于 GPU 加速的高吞吐多度量空间并发数据挖掘, 提升算法性能; (3) 基于度量学习的多度量空间数据聚类加速, 通过智能化的学习型方法降低距离计算开销, 进一步提升查询性能.

References:

[1] Ester M, Kriegel HP, Sander J, Xu XW. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proc. of the 2nd Int'l Conf. on Knowledge Discovery and Data Mining. Portland: AAAI Press, 1996. 226−231.

[2] Yu YW, Jia ZF, Cao L, Zhao JD, Liu ZW, Liu JL. Fast density-based clustering algorithm for location big data. Ruan Jian Xue Bao/Journal of Software, 2018, 29(8): 2470–2484 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5289.htm [doi: 10.13328/j.cnki.jos.005289]

[3] Gunasekaran YD, Rahman MF, Hasani S, Zhang N, Das G. DBLOC: Density based clustering over location based services. In: Proc. of the 2018 Int'l Conf. on Management of Data. Houston: ACM, 2018. 1697−1700. [doi: 10.1145/3183713.3193561]

[4] Kim B, Koo K, Kim J, Moon B. DISC: Density-based incremental clustering by striding over streaming data. In: Proc. of the 37th IEEE Int'l Conf. on Data Engineering. Chania: IEEE, 2021. 828−839. [doi: 10.1109/ICDE51399.2021.00077]

[5] Yang D, Rundensteiner EA, Ward MO. Summarization and matching of density-based clusters in streaming environments. Proc. of the VLDB Endowment, 2011, 5(2): 121−132. [doi: 10.14778/2078324.2078328]

[6] Zhang BX, Zhang LF, Wang Z, Cui ZQ, Sun Y, Hua HC. Image reconstruction of planar electrical capacitance tomography based on DBSCAN and self-adaptive ADMM algorithm. IEEE Trans. on Instrumentation and Measurement, 2023, 72: 4504711. [doi: 10.1109/TIM.2023.3284931]

[7] Ren YZ, Wang N, Li MX, Xu ZL. Deep density-based image clustering. Knowledge-based Systems, 2020, 197: 105841. [doi: 10.1016/j.knosys.2020.105841]

[8] Al-Shammari A, Zhou R, Naseriparsaa M, Liu CF. An effective density-based clustering and dynamic maintenance framework for evolving medical data streams. Int'l Journal of Medical Informatics, 2019, 126: 176–186. [doi: 10.1016/j.ijmedinf.2019.03.016]

[9] Chu YW, Luo XB, Qu K, Tao YB, Lin J, Lin H. DBSCAN and *K*-means hybrid clustering based automatic dental feature detection. Journal of Computer-Aided Design & Computer Graphics, 2018, 30(7): 1276–1283 (in Chinese with English abstract). [doi: 10.3724/SP.J.1089.2018.16736]

[10] Yang KY, Gao YJ, Ma R, Chen L, Wu S, Chen G. DBSCAN-MS: Distributed density-based clustering in metric spaces. In: Proc. of the 35th IEEE Int'l Conf. on Data Engineering. Macao: IEEE, 2019. 1346−1357. [doi: 10.1109/ICDE.2019.00122]

[11] Chen L, Gao YJ, Zhang YL, Jensen CS, Zheng BL. Efficient and incremental clustering algorithms on star-schema heterogeneous graphs. In: Proc. of the 35th IEEE Int'l Conf. on Data Engineering. Macao: IEEE, 2019. 256−267. [doi: 10.1109/ICDE.2019.00031]

[12] Chen L, Gao YJ, Huang XR, Jensen CS, Zheng BL. Efficient distributed clustering algorithms on star-schema heterogeneous graphs. IEEE Trans. on Knowledge and Data Engineering, 2020, 34(10): 4781–4796. [doi: 10.1109/TKDE.2020.3047631]

[13] Gunawan A. A faster algorithm for DBSCAN [MS. Thesis]. Eindhoven: Eindhoven University of Technology, 2013.

[14] Gan JH, Tao YF. DBSCAN revisited: Mis-claim, un-fixability, and approximation. In: Proc. of the 2015 ACM SIGMOD Int'l Conf. on Management of Data. Melbourne: ACM, 2015. 519−530. [doi: 10.1145/2723372.2737792]

[15] Lulli A, Dell'Amico M, Michiardi P, Ricci L. NG-DBSCAN: Scalable density-based clustering for arbitrary data. Proc. of the VLDB Endowment, 2016, 10(3): 157–168. [doi: 10.14778/3021924.3021932]

[16] Moseley B, Vassilvitskii S, Wang YY. Hierarchical clustering in general metric spaces using approximate nearest neighbors. In: Proc. of the 24th Int'l Conf. on Artificial Intelligence and Statistics. PMLR, 2021. 2440−2448.

[17] Li H, Liu XJ, Li T, Gan RD. A novel density-based clustering algorithm using nearest neighbor graph. Pattern Recognition, 2020, 102: 107206. [doi: 10.1016/j.patcog.2020.107206]

[18] Chen YW, Zhou LD, Pei SW, Yu ZW, Chen Y, Liu X, Du JX, Xiong NX. KNN-BLOCK DBSCAN: Fast clustering for large-scale data. IEEE Trans. on Systems, Man, and Cybernetics: Systems, 2021, 51(6): 3939–3953. [doi: 10.1109/TSMC.2019.2956527]

[19] Yu YX, Wang GR, Lin LZ, Li M, Zhu XH. M^{2+}-Tree: Processing multiple metric space queries of medical cases efficiently with just one index. Journal of Computer Research and Development, 2010, 47(4): 671–678 (in Chinese with English abstract).

[20] Bustos B, Skopal T. Dynamic similarity search in multi-metric spaces. In: Proc. of the 8th ACM Int'l Workshop on Multimedia Information Retrieval. Santa Barbara: ACM, 2006. 137−146. [doi: 10.1145/1178677.1178698]

[21] Franzke M, Emrich T, Züfle A, Renz M. Indexing multi-metric data. In: Proc. of the 32nd IEEE Int'l Conf. on Data Engineering. Helsinki: IEEE, 2016. 1122−1133. [doi: 10.1109/ICDE.2016.7498318]

[22] Zhu YF, Chen L, Gao YJ, Zheng BH, Wang PF. DESIRE: An efficient dynamic cluster-based forest indexing for similarity search in multi-metric spaces. Proc. of the VLDB Endowment, 2022, 15(10): 2121–2133. [doi: 10.14778/3547305.3547317]

[23] Zabot GF, Cazzolato MT, Scabora LC, Traina AJM, Traina C. Efficient indexing of multiple metric spaces with spectra. In: Proc. of the 2019 IEEE Int'l Symp. on Multimedia. San Diego: IEEE, 2019. 169−1697. [doi: 10.1109/ISM46123.2019.00038]

[24] Liu JL, Sun HL. Finding mutual k-nearest neighbors in multi-metric space. Journal of Frontiers of Computer Science & Technology, 2010, 4(10): 881–889 (in Chinese with English abstract). [doi: 10.3778/j.issn.1673-9418.2010.10.002]

[25] Chen L, Gao YJ, Song X, Li Z, Zhu YF, Miao XY, Jensen CS. Indexing metric spaces for exact similarity search. ACM Computing Surveys, 2022, 55(6): 128. [doi: 10.1145/3534963]

[26] Zhu YF, Chen L, Gao YJ, Jensen CS. Pivot selection algorithms in metric spaces: A survey and experimental study. The VLDB Journal, 2022, 31(1): 23–47. [doi: 10.1007/s00778-021-00691-4]

[27] Micó ML, Oncina J, Vidal E. A new version of the nearest-neighbour approximating and eliminating search algorithm (AESA) with linear preprocessing time and memory requirements. Pattern Recognition Letters, 1994, 15(1): 9–17. [doi: 10.1016/0167-8655(94)90095-7]

[28] Bozkaya T, Ozsoyoglu ZM. Distance-based indexing for high-dimensional metric spaces. ACM SIGMOD Record, 1997, 26(2): 357–368. [doi: 10.1145/253262.253345]

[29] Chávez E, Navarro G. A compact space decomposition for effective metric indexing. Pattern Recognition Letters, 2005, 26(9): 1363–1376. [doi: 10.1016/j.patrec.2004.11.014]

[30] Ciaccia P, Patella M, Zezula P. M-tree: An efficient access method for similarity search in metric spaces. In: Proc. of the 23rd Int'l Conf. on Very Large Data Bases. Athens: Morgan Kaufmann Publishers Inc., 1997. 426−435.

[31] Brin S. Near neighbor search in large metric spaces. In: Proc. of the 21st Int'l Conf. on Very Large Data Bases. Zurich: Morgan Kaufmann Publishers Inc., 1995. 574−584.

[32] Novak D, Batko M, Zezula P. Metric index: An efficient and scalable solution for precise and approximate similarity search. Information Systems, 2011, 36(4): 721–733. [doi: 10.1016/j.is.2010.10.002]

[33] Skopal T, Pokorný J, Snásel V. PM-tree: Pivoting metric tree for similarity search in multimedia databases. In: Proc. of the 8th Advances in Databases and Information Systems. Budapest: ADBIS, 2004. 803−815.
[34] Bentley JL. Multidimensional binary search trees used for associative searching. Communications of the ACM, 1975, 18(9): 509–517. [doi: 10.1145/361002.361007]
[35] Xu XW, Jäger J, Kriegel HP. A fast parallel clustering algorithm for large spatial databases. Data Mining and Knowledge Discovery, 1999, 3(3): 263–290. [doi: 10.1023/A:1009884809343]
[36] Wang MZ, Xu XL, Yue Q, Wang YX. A comprehensive survey and experimental comparison of graph-based approximate nearest neighbor search. Proc. of the VLDB Endowment, 2021, 14(11): 1964–1978. [doi: 10.14778/3476249.3476255]
[37] Malkov Y, Ponomarenko A, Logvinov A, Krylov V. Approximate nearest neighbor algorithm based on navigable small world graphs. Information Systems, 2014, 45: 61–68. [doi: 10.1016/j.is.2013.10.006]
[38] Open food facts. 2023. https://world.openfoodfacts.org/data
[39] Popani. Air quality data in India (2015–2020). 2020. https://www.kaggle.com/datasets/rohanrao/air-quality-data-in-india
[40] Two sigma connect: Rental listing inquiries. 2023. https://www.kaggle.com/c/two-sigma-connect-rental-listing-inquiries
[41] Content-based photo image retrieval test-collection. 2023. http://cophir.isti.cnr.it
[42] Geographical locations in los angeles. 2023. https://www.dbs.ifi.lmu.de/cms
[43] Moby project. 2023. https://mobyproject.org
[44] Word to vectors. 2023. https://code.google.com/archive/p/word2vec
[45] Fritz M, Behringer M, Tschechlov D, Schwarz H. Efficient exploratory clustering analyses in large-scale exploration processes. The VLDB Journal, 2022, 31(4): 711–732. [doi: 10.1007/s00778-021-00716-y]
[46] Rousseeuw PJ. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 1987, 20: 53–65. [doi: 10.1016/0377-0427(87)90125-7]

附中文参考文献:

[2] 于彦伟, 贾召飞, 曹磊, 赵金东, 刘兆伟, 刘惊雷. 面向位置大数据的快速密度聚类算法. 软件学报, 2018, 29(8): 2470–2484. http://www.jos.org.cn/1000-9825/5289.htm [doi: 10.13328/j.cnki.jos.005289]
[9] 褚玉伟, 罗晓博, 屈珂, 陶煜波, 林军, 林海. DBSCAN 和 K-means 混合聚类的牙齿特征自动识别. 计算机辅助设计与图形学学报, 2018, 30(7): 1276–1283. [doi: 10.3724/SP.J.1089.2018.16736]
[19] 于亚新, 王国仁, 林利增, 李淼, 朱歆华. M^{2+}-树: 一种支持医学病例多度量空间检索的高效索引. 计算机研究与发展, 2010, 47(4): 671–678.
[24] 刘俊岭, 孙焕良. 多维度量空间中发现相互 kNN. 计算机科学与探索, 2010, 4(10): 881–889. [doi: 10.3778/j.issn.1673-9418.2010.10.002]

朱轶凡(1997－), 男, 博士, 主要研究领域为大数据管理与分析.

陈璐(1989－), 女, 博士, 研究员, 博士生导师, CCF 专业会员, 主要研究领域为度量空间数据管理, 时空数据管理, 查询可用性分析.

罗程阳(1998－), 男, 硕士生, 主要研究领域为数据库可用性, 社区发现.

毛玉仁(1991－), 男, 博士, 研究员, 博士生导师, CCF 专业会员, 主要研究领域为机器学习, 智能数据库.

马瑞遥(2000－), 女, 硕士生, 主要研究领域为多模态数据检索.

高云君(1977－), 男, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究领域为数据库, 大数据管理与分析, DB 与 AI 融合.