第34卷 第10期 计 算 机 工 程 2008年5月
Vol.34 No.10 Computer Engineering May 2008
·人工智能及识别技术· 文章编号：1000—3428(2008)10—0190—03 文献标识码：A 中图分类号：TP391

适用于公交站点聚类的DBSCAN改进算法

蔡永旺，杨炳儒
(北京科技大学信息工程学院，北京 100083)

摘 要：提出一种适用于公交站点聚类的DBSCAN改进算法，缩小搜索半径ε，从而提高聚类正确度，同时通过共享对象判定连接簇的合并，防止簇的过分割，减少噪声点，有效地屏蔽了算法对输入参数的敏感性，提高聚类结果的质量，减少密度差距对聚类结果的影响。保持DBSCAN算法的高执行效率，并应用在智能公交换乘查询引擎中公交站点聚类，聚类准确率提高了16%，验证了新算法的有效性。
关键词：聚类；DBSCAN算法；参数敏感；数据挖掘

Improved DBSCAN Algorithm for Public Bus Station Cluster

CAI Yong-wang, YANG Bing-ru
(School of Information, Beijing University of Science and Technology, Beijing 100083)

【Abstract】 The paper presents an improved DBSCAN algorithm for bus station cluster, which improves the degree of accuracy by shrinking the searching radius and merging the clusters by the share object. As a result it efficaciously avoids over-segmentation and reduces the number of noise points. Thus, it effectively shields the sensitivity of the input parameter, produces better clustering results, and reduces the influence to the cluster result by density disparity. And the high performance of the origin algorithm is maintained at the same time. Experimental results demonstrate by cluster of Beijing bus station. And the accuracy of cluster increasesby 16%, which indicates that improve algorithm is valid.
【Key words】 clustering; Density Based Spatital Clustering of Application with Noise(DBSCAN) algorithm; sensitive to input parameter; data mining

1 概述

在城市智能公交换乘查询引擎的研究与设计中，主要的研究对象是公交车站和公交车线，而引擎设计的主要难题也在于公交车站的处理上，因为在一个城市中公交车线往往有限，最多不过千余条，而公交车站则是其几十倍，往往数以万计，如果直接对其操作，势必要消耗大量计算机资源，降低引擎的响应时间。但这些公交车站并不是随机分布的，它们往往在地理位置上相近，如果将这些地理位置相近的归结为一个车站，给它们同一个ID，将大幅减少公交车站的数量，方便智能公交换乘查询引擎的设计。

聚类分析提供了解决上述问题的办法，它自动根据相似性对数据对象进行分组，发现数据空间的分布特征，是数据挖掘的一类主要方法。既可以独立使用，也可以作为其他数据挖掘算法的预处理步骤。

聚类是找出样本比较密集的部分，每一个密集部分就是一个类。从这个角度出发，就可设计一个密度函数，计算出每个样本附近的密度，从而根据每个样本附近的密度值来找出那些样本相对比较集中的区域。

2 DBSCAN算法

DBSCAN算法是一种颇具代表性的基于密度的聚类分析算法，它可以发现任意形状的簇和有效屏蔽噪声数据干扰。

2.1 算法思想[1-2]

DBSCAN算法利用类的高密度连通性，快速发现任意形状的类，其核心思想是：对于构成簇的每个对象，其ε邻域包含的对象个数，必须不小于一个给定值*MinPts*，即对于一个类中的每一个对象，在其给定半径的邻域内包含的对象数不能少于某一个给定的最小数目。下面是基于密度的聚类的基本思想涉及的一些定义：

(1)给定对象半径ε内的区域称为该对象的ε-领域。

(2)如果一个对象的ε-领域至少包含最小数目*MinPts*个对象，则称该对象为核心对象。

(3)给定一个对象集合D，如果p是在q的ε-领域内，而q是一个核心对象，则称对象p从对象q出发是直接密度可达的。

(4)如果存在一个对象链$p_1,p_2,\cdots,p_n,p_1=q,p_n=q$，对$p_i\in D,1\le i\le n$，$p_{i+1}$是从$\varepsilon$关于$p_i$和*MinPts*直接密度可达的，则称对象$p$从对象$q$关于$\varepsilon$和*MinPts*密度可达的(density-reachable)，这种关系是非对称关系。

(5)如果对象集合D中存在一个对象o，使得对象p和q是从o关于ε和*MinPts*密度可达的，对象p和q是关于ε和*MinPts*密度相连的(density-connected)，这种关系是对称关系。

(6)一个基于密度的簇是基于密度可达性的最大的密度相连对象的集合。不包含在任何族中的对象被称为噪声。

2.2 算法内容[1-2]

基于上述定义，DBSCAN算法采用了一种查找密度可达对象的方法进行聚类分析。DBSCAN算法的正确性由下述事实保证：一个簇与由簇中任意核心对象密度可达的所有对象

基金项目：国家自然科学基金资助项目(60675030)
作者简介：蔡永旺(1982－)，男，助教、硕士研究生，主研方向：人工智能，数据挖掘；杨炳儒，教授、博士生导师
收稿日期：2007-05-25 **E-mail：**caiyw@126.com

构成的集合 O是等价的。DBSCAN 算法采用迭代查找的方法，通过迭代地查找所有直接密度可达的对象，找到各个簇所包含的所有密度可达的对象。具体方法如下：

(1)检查数据库中尚未检查过的对象 p，如果 p 未被处理(归入某个簇或标记为噪声)，则检查其 ε 邻域 $N_\varepsilon(p)$，若包含的对象数不小于 $MinPts$，则建立新簇 C，将 $N_\varepsilon(p)$ 中所有点加入 C；

(2)对 C 中所有未被处理的对象 q，检查其 ε 邻域 $N_\varepsilon(p)$，若 $N_\varepsilon(q)$ 包含至少 $MinPts$ 个对象，则将 $N_\varepsilon(p)$ 中未归入任何一个簇的对象加入 C；

(3)重复步骤(2)继续检查 C 中未处理对象，直到没有新的对象加入当前簇 C；

(4)重复步骤(1)～步骤(3)，直到所有对象都归入了某个簇或标记为噪声。

密度可达对象的获取是通过不断执行区域查询来实现一个区域查询返回指定区域中的所有对象。

2.3 DBSCAN 的主要不足及已有改进算法

DBSCAN算法有两个主要缺点[3-4]：(1)对输入参数ε敏感，只能发现密度相近的类，影响了聚类结果的质量。确定邻域大小的参数ε是由用户指定的，但是实际上参数ε很难在算法运行前确定。(2)难以发现密度相差较大的簇。由于决定密度阈值的参数ε和$MinPts$是全局唯一的，DBSCAN只能发现密度近似的簇。

目前主要改进算法有OPTICS算法[2]与PDBSCAN算 法[3-5]，这两种算法虽然能较好地解决了这两个缺点，但是由于采用了复杂的处理方法以及额外的磁盘 I/O操作，使得运行速度远远低于DBSCAN。

3 屏蔽参数敏感的 MDBSCAN 算法

本文针对 DBSCAN 算法的这两个缺点 ，结合公交站点的特点，提出了一种 DBSCAN 的改进算法 MDBSCAN。

3.1 算法思想

公交站点难于归类处理，主要表现为以下特征：(1)车站名称相同但是地理位置不同；(2)车站名称不同但是地理位置相同；(3)既不同站也不同名，但是地理位置相近。为了保证公交线路换乘查询引擎计算结果的有效性，将这 3 类车站有效、准确地进行聚类是关键。为了解决以上问题，本算法改变了以往一个簇一个 ID 的方法，改用一个簇赋值两个类型 ID 的方法，簇 ID(ID_Cluster)和信息 ID(ID_Info)，ID_cluster 用来记录聚类结果，其作用与 DBSCAN 中的 ID 一致，ID_Info 记录站点原有信息，为以后取出站点信息所用，解决聚类站点信息错误的问题。

这几个问题产生的根本原因是邻近公交车站之间的交叉站点造成的，如北京市阜成门地区，200 m 内分布有阜成门内、阜成门、阜成门外 3 个车站，每个车站又有多条车线通过，按本引擎所用数据集的数据采集方式，造成了这 200 m 内连续分布有 50 个站点，这些连续点分属于这 3 个车站且彼此相距很近，连为一体，很难将其准确聚类，即算法的核心内容要就是对共享对象的处理问题。所谓共享对象 p，是指数据库中若存在至少 2 个核心对象 o_1,o_2，对象 p 分别输入 o_1,o_2，且 p 分别从 o_1,o_2 密度可达，如图 1 所示。如果簇 $C1$ 和簇 $C2$ 有共享对象，则称簇 $C1$ 和 $C2$ 相通的。首先，为了有效地执行区域查询，MDBSCAN 算法使用了空间查询中的 R^* 树结构。在进行聚类前，必须建立针对所有数据的 R^* 树，使算法空间复杂度变为 $O(\mathrm{lb}n)$。其次，算法中需要指定一个全局参量半径 ε。为了确定 ε 值，MDBSCAN 通过统计学的方法，对所有小规模公交车站进行抽样，进过计算，得到一般小规模的公交车站大约长 25 m，每个车站约有 8 条车线，所以规定初始值 $MinPts=8$，$\varepsilon=25$。

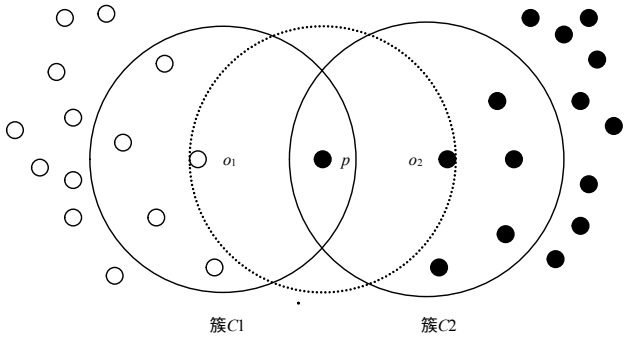

图 1 共享对象示意图

MDBSCAN 算法中，在发现了共享对象时，验证共享对象 p 在数据库中的其余属性是否与核心对象 o 一致，如果一致则将其归入 o 所在的簇 c，并记录下来， 供迭代查找过程结束后进一步处理。否则不标记，继续查找下一个对象。待所有点处理完毕后，如果连接簇的共享对象超过一定数量 m(经过抽样计算，在本文取 $m=4$)，则说明这两个簇的地理位置非常接近，可以进行合并，簇合并的实质，是在 2 个簇共享的连接对象处，降低了密度可达阈值 $MinPts$，使得两个簇的核心对象密度可达。簇合并的结果体现在簇 ID 的更新上，而不更新信息 ID，以保持各簇原有的信息，防止站点信息错误。

3.2 算法描述

MDBSCAN 算法的主要改进是在 DBSCAN 算法的 ExpandCluster 函数中 ，下面为改进后的 ExpandCluster 函数，其 C 语言伪码描述为

```
Bool ExpandCluster(SetOfPoints, Point, ClId, ε, Minpts)
{
seeds=SetOfPoints.regionQuery(Point, ε);
 //返回集合 SetOfPoints 中点 Point 的所有 ε-相邻点
     if (seeds.size<Minpts) //不是核心点
     {
      SetOfPoints.changeClId(Point, NOISE); //标记为噪声点
       return false; }
      else   //seeds 中所有的点从点 Point 都密度可达
     {
       SetOfPoints.ChangeClIds(seeds, ClId); //标记类 id 为 ClId
       if(!isshare(point)&&checkinfo(point))
         //如果不是共享对象且信息不统一
         seeds.delete(Point);
       else
       savesharep(point, ID_info)//保存共享对象信息
       while(seeds!=null)
       {
         currentp=seeds.first();
         result=SetOfPoints.regionQuery(currentp, ε);
         if (result.size>=Minpts)
         {
            for (int i=1; i<=result.size; i++)
  { resultp=result.get(i); //取得 result 集合中的第 i 个点
  if (resultp.ClId In {UNCLASSIFIED, NOISE})
```

```
    //若第 i 个点未分类或者是噪声点
    {
        if (resultp.ClId=UNCLASSIFIED)
        { //未分类
        seeds.append(resultp); //加入 seends 集合
        SetOfPoints.changeClId(resultp, ClId);//改变其类的 id} }
    seeds.delete(currentp); //删除 seeds 集合中的当前点} } }
  if(sharepoint > 4)
  combincluster(Id_cluster); //合并共享对象>4 的连接对象}
```

4 算法性能测试

笔者对算法 MDBSCAN 的性能进行了测试，并将测试结果与经典 DBSCAN 做了比较，所有测试是在同一台 PC 机上进行的，配置为 PM 1.4 GHz CPU，512 MB 主存，80 GB 硬盘，算法用 C++语言实现，算法数据集采用的是存储于 Mapinfo 7 数据库的北京市所有公共汽车及地铁站点，共计约 95 000 个，聚类完毕后，共计 6 531 个公交车站，与北京实际站点数相符。测试结果如图 2 所示，准确率对比如表 1 所示，西直门地区车站聚类前后效果如图 3、图 4 所示。图中，★表示核心对象。

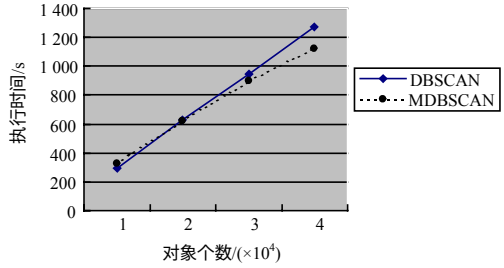

图 2　MDBSCAN 与 DBSCAN 性能比较

表 1　MDBSCAN 与 DBSCAN 准确率比较

	抽取站点数	正确站点数	准确率/(%)
DBSCAN	197	141	71
MDBSCAN	197	173	87

图 3　西直门地区聚类前站点

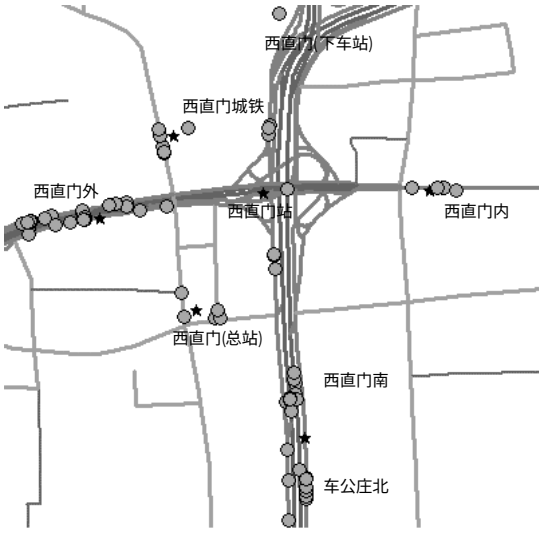

图 4　西直门地区聚类后站点

5 结束语

本文分析了 DBSCAN 算法对输入 ε 参数敏感的问题，讨论了近年来已有的改进算法，提出一种新的改进算法 MDBSCAN。MDBSCAN 在保持原 DBSCAN 算法较高执行效率的同时，有效地解决了算法对输入参数敏感的问题。通过智能公交查询引擎的实际应用表明，MDBSCAN 算法的执行效率与 DBSCAN 算法接近，车站聚类的准确度比 DBSCAN 高，算法有效地解决了参数 ε 值过小时的聚类结果错误的问题，同时再通过共享对象判定簇的连接，也就是小块车站的连接，保证了聚类结果的高正确率。

目前，本算法的聚类准确度还不是很高，在对公交站点数据进行聚类后，还需要对结果做一些检验复查工作，在一些复杂站点上需要人工干预。作者下一步的工作是进一步提高聚类结果的准确性，减少人工干预。

参考文献

[1] Ester M, Krieger H P, Sander J, et al. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise[C]//Proc. of the 2nd Int'l Conf. on Knowledge Discovery and Data Mining. [S. l.]: ACM Press, 1996.

[2] Han Jiawen, Kambr M. Data Mining Concepts and Techniques[M]. 北京: 高等教育出版社, 2001.

[3] 周水庚, 周傲英, 曹　晶. 基于数据分区的 DBSCAN 算法[J]. 计算机研究与发展, 2000, 37(10): 1153-1159.

[4] 周水庚, 周傲英, 金　文, 等. FDBSCAN: 一种快速 DBSCAN 算法[J]. 软件学报, 2000, 11(6): 735-744.

[5] Guan Jihong, Zhou Shuigeng, Bian Fuling, et al. Scaling up the DBSCAN Algorithm for Cluster in large Spatial Databases Based on Sampling Technique[J]. Wuhan University Journal of Natural Sciences, 2001, 6(1/2): 467-473.