

一种基于参考点的快速密度聚类算法

闫 安,刘琪林

(合肥工业大学 管理学院,安徽 合肥 230000)

摘 要:提出了一种基于 k 个参考点 DCUR (Density-based Clustering Using k References). 新算法以 k 个参考点反应了数据的分布状态,然后基于参考点对数据进行聚类分析. 该算法保持了 DBSCAN 的优点,并且可以减少区域查询次数,降低 I/O 开销. 理论和实验证明新算法能够有效地对大规模数据库进行聚类,且其执行效率明显高于传统的基于 R^* 树的 DBSCAN 算法.

关键词: 聚类;密度;簇;参考点;数据挖掘

中图分类号: TP301

文献标识码: A

文章编号: 1000—7180(2017)10—0032—04

A Fast Density Clustering Algorithm Based on Reference

YAN An,LIU Qi-lin

(Hefei University of Technology Anhui Province, School of Management, Hefei 230000, China)

Abstract: In this paper, a new kind of clustering algorithm that is called DCUR (density-based clustering using k references) is presented. Innovative point is that the new algorithm to k references response data Distribution, and then analyzes the data based on the k references. DCUR keeps the ability of density based clustering method's good features, and it can reach high efficiency and the execution frequency of region query can be decreased, and consequently the I/O cost is reduced, so the new algorithm can reach high efficiency and reduce the complexity of the algorithm. Both theory analysis and experimental results confirm that DCUR is effective and efficient in clustering large-scale database, and its executing efficiency is much higher than traditional DBSCAN algorithm based on R^* -tree.

Key words: clustering; density; cluster; reference; data mining

1 引言

作为数据挖掘和人工智能领域的重要研究课题,聚类(Clustering)是一种无监督学习. 数据的表示、数据的元素间相似度的表示以及数据元素分组的不同技术产生了许多聚类分析方法,如基于划分的算法 (K-MEANS^[1]、K-MEDOIDS^[2]、CLARANS)、基于层次的算法 (BIRCH^[3]、CURE)、基于密度的算法 (DBSCAN^[4]、DENCLUE、OPTICS^[5])、基于网格的算法 (STING、CLIQUE)、基于模型的方法 (统计学^[6]、神经网络^[7])等,同时对于聚类算提出有效性研究要求^[8]. 基于密度的聚类

方法中 DBSCAN 是聚类中一种典型算法,该算法可以挖掘任意形状的聚类,对数据输入顺序不敏感,并且具有处理异常数据的能力,但是时间复杂度较高,在空间索引如 R^* 树的支持下,其复杂性为 $O(N\log N)$. 于是研究者在 DBSCAN 基础上进行改进和实现^[9],提出了如 IDBSCAN^[10]、CURD 等,大大地提高了聚类时间和质量.

本文以 DBSCAN 为基础,提出了一种基于 k 个参考点 DCUR (Density-based Clustering Using k References). 该算法与 DBSCAN 算法不同的是 DUCR 以 k 个参考点代表所有核心点对数据进行聚类,并且同时找出 k 个参考点之间的关系进行簇

的分析,这样就直接地提高了算法的效率,降低计算机的存储空间. 为了更加方便地描述 DCUR 算法,本文以二维空间为例来分析该算法.

2 DCUR 聚类算法基本概念

本文算法在 DBSCAN 的基础上对一些相关术语做出了定义. 以下定义涉及到两个参数:空间点邻域范围半径 Eps(Epsilon neighborhood)和最小邻居点阈值 MinPts(Minimum number of points).

定义 1 核心点:对于给定的密度阈值 Minpts,如果点 p 的密度不低于 MinPts,则称该点为核心点,反之为非核心点.

定义 2 参考点:代表若干核心点的集合的核心点为参考点. 参考点在一定的程度表现出了若干个核心点的分布情况.

定义 3 邻参考点:给定距离 Eps 和密度阈值 MinPts,如果参考点参考点 p, q 满足 $\text{Dist}(p, q) \leq 2 \cdot \text{Eps}$, 即称 p, q 为邻参考点.

定义 4 欧氏距离 (euclidean distance): $d_{ij} = \sqrt{\sum_{k=1}^p (x_{ik} - x_{jk})^2}$ 本文采取的距离都是欧式距离.

定义 5 簇 (cluster):将空间数据点集合 D 中所有密度相连的空间点聚集成多个子集合,每个非空子集称为一个聚类 C . 对每个簇 C 来说都可以表示由其中的一个参考点及密度可达于该参考点的所有空间点构成.

3 DCUR 聚类算法

DCUR 的算法框架如图 1 所示. 首先寻找输入数据集 D 中所有核心点;接着寻找能够代表若干核心点分布状态的 k 个参考点;然后分类 k 个参着考点,每个类构成了一个聚类的基本信息;最后处理个别点 (边界点和噪声点),形成一个聚类.

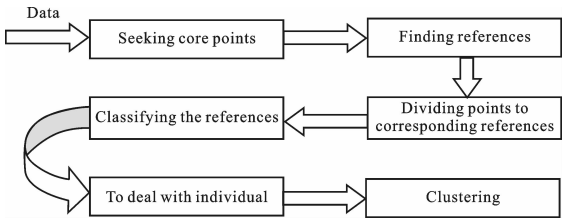


图 1 DCUR 算法

3.1 寻找 k 个参考点

寻找思想. 在 DBSCAN 算法中,每个核心点可以代表着一个类,因此要想更快更好的得到聚类结果,可以选择 k 个参考点能够代表所有核心点的分

布信息来进行聚类,同时,这些参考点的距离要尽可能大但又不失代表性. 鉴于此,对于参考点的选择,着眼于如何使它们之间的距离尽可能大. 考虑到数据集往往比较大,采用逐一划分的方法得到聚类中心即参考点. 首先查询得到所有的核心点,接着选取相距最远的两个核心点作为最初的两个参考点. 以这两个参考点为依据,将剩余的参考点按距离最小划分到离它最近的参考点所属的类别中. 待划分完毕,找出包含核心点最多的一个类,再在这个类别中找出距离最大的两个核心点作为下两个参考点. 然后再以这两个参考点为对象进行划分. 以此类推,不断划分包含核心点最多的一个类别寻找参考点,直到找到 k 个参考点为止. 由于不断划分数据集的核心点,并将最远的核心点作为参考点. 因此通过此种方法获得 k 个参考点任意两个之间都会有较大的距离.

k 的取值界限:当以某个参考点包含最多的核心点的类别中,选取的两个距离最大的核心点的距离不大于 Eps 时,则不需要再寻找参考点, k 值取值结束.

寻找参考点的具体步骤如下:

- (1) 预处理数据集 D
- (2) 查询数据集 D 中核心点的数量 M ,将此 M 作为一个类别 C_0 .
- (3) 计算 C_0 中两两核心点之间的距离,找出距离最大点 x_1, x_2 作为最初两个参考点,并相应地建立类别 C_1, C_2
- (4) 分别计算 C_0 中其他核心点到参考点 x_1, x_2 的距离,将它们划分到离两个参考点中距离最近的参考点的类别 C_1, C_2 中去
- (5) 取含核心点最多的类别,重复步骤(3)(4)
- (6) 判断 k 值大小:当某个含最多核心点的类别中距离最大的两个核心点之间的距离不大于 Eps, 则结束

算法的伪代码如下:

```
class Point
{
public:
    friend float distance(Point &left, Point &right);
private:
    float x;
    float y;
};
float distance(Point &left, Point &right)
{
    return ((left.x - right.x) ^ 2 + (left.y - right.y) ^ 2)
```

```

2);
}
vector<Point> C0, C1, C2;
void find(vector<Point> &D)
{
    Point p1, p2;
    float temp=0;
    for (auto k : D)
    {
        if (k == core Point)
        {
            C0.push_back(k);
        }
    }
    for (int i = 0; i < C0.size(); i++)
    {
        for (int j = i; j < C0.size(); j++)
        {
            if (distance(C0[i], C0[j]) > temp)
            {
                temp = distance(C0[i], C0[j]);
                p1 = C0[i];
                p2 = C0[j];
            }
        }
    }
    C1.push_back(p1);
    C2.push_back(p2);
}
void deuide(vector<Point> &C0, vector<Point> &C1,
vector<Point> &C2)
{
    for (auto c : C0)
    {
        if (distance(c, C1.at(0)) < distance(c, C2.at(1)))
        {
            C1.push_back(c);
        }
        else
        {
            C2.push_back(c);
        }
    }
}

```

```

int main()
{
    vector<Point> D, C0, C1, C2;
    DInit(D); //初始化数据集 D
    find(D); //查询数据集中核心点的数量 M
    deuide(C0, C1, C2); //分别计算核心点到 x1, x2 的
    距离,将最近的点放到 C1, C2 中
}

```

寻找参考点中有两个函数处理过程:(1)Calculate_point: 计算最大聚类中相距最远的核心点作为

参考点;(2)Min_distance: 将剩余的核心点划分到离当前参考点最近的类别当中. DCUR 算法其实就是这两个函数的有限次重复的过程. 下面具体介绍这两个函数处理过程,如图 2、图 3 所示.

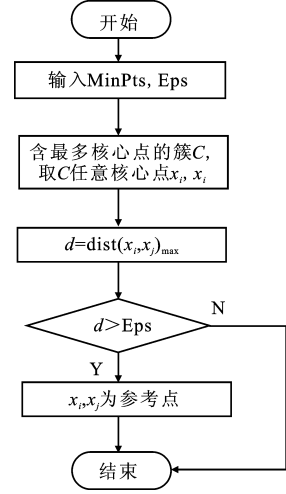


图 2 Calculate_point 函数处理过程

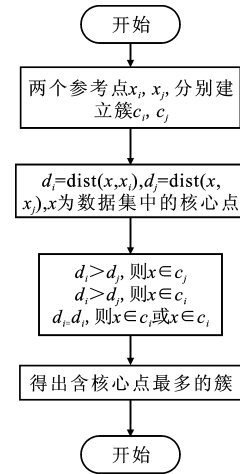


图 3 Min_distance 函数处理过程

3.2 参考点的分类

定理 1 若一个参考点 p 同时属于两个簇 C_1 和 C_2 , 则 C_1 和 C_2 可密度相连为一个簇 C .

证明: 数据集合为 D , 因为核心点 $p \in C_1$, 从聚类簇的概念可得, 任意的属于簇 C_1 的点 x_1 都密度相连于点 p , 因点 p 为 C_1 的核心点, 所以任意点 $x_1 (x_1 \in C_1)$ 都密度可达与点 p . 同理可知, 对于任意点 $x_2 (x_2 \in C_2)$ 也都密度可达与点 p . 综上可知, 任意的两个点 $x_1, x_2 (x_1 \in C_1, x_2 \in C_2)$ 都密度相连, 所以 C_1, C_2 可密度相连为一个簇. 因此, 若参考点 p 同时属于邻参考点所属的聚类簇中, 则我们就可以合并这些簇为一个簇. 同时也得出参考点最终

仅属于一个簇。

3.3 建立点、参考点、聚类的映射

定理 2 边界点至少属于一个簇,而噪声点不属于任何簇.此定理可以用来对遗漏点进行簇的判断.

参考点的代表区域其实已经就建立了参考点与数据点之间的对应关系,边界点和噪声点的处理也让此对应关系更加清楚明了.同时对参考点的分类过程中实际上已经建立了聚类 and 参考点之间的映射,属于同一个分类中的参考点的代表区域的输入数据的集合就是一个聚类,所以聚类,参考点,数据之间的映射非常简单明了.

3.4 DCUR 的时空复杂度的分析

假定数据量为 N ,在寻找参考点的过程中,核心点的数目为 M ,循环次数为 I ,参考点数目为 K ,聚类的个数为 C . n 是聚类后数据集中剩余的未被聚到任何簇中的点数.其中 $n \ll N$, I 通常也很小.而整个数据集的密度级别是远远小于 N 的.因此容易知道 DCUR 的时间复杂度近似为 $O(I \cdot M \cdot N + K \cdot N) + O(C \cdot I \cdot n)$, DCUR 空间复杂度为 $O(N) + O(M) + O(C)$.

4 DCUR 的算法测试

本文对 DCUR 算法的性能进行测试,并且将测试的结果与 DBSCAN 进行比较,本文选择将测试的结果与 DBSCAN 比较的主要原因有三点:(1)新算法是基于 DBSCAN 基础上创新而来,都是基于密度的聚类算法;(2)这两种算法都需要设置 Eps、MinPts 两种参数;(3)DBSCAN 在建立 $R \times$ 树下时间复杂度为 $O(N \log N)$,因此于之比较才会有客观认识.算法的所有测试均在一台普通 PC 机(P4 2.3 GHz CPU、512 MB 内存、80 GB 硬盘)上进行,算法用 MATLAB6.5 实现.算法的测试数据集同时使用真实数据(SEQUOIA2000 数据库,该数据库也被文献[4]用来测试 DBSCAN 的算法性能)和自行构造的模拟数据的两种数据进行测试.

可以看出,DCUR 算法与 DBSCAN 算法随着数据量的增加呈现类指数增长,并且 DCUR 比 DBSCAN 算法快的多,随着数据量的增大,这种时间速度上的差距更加的明显.可以看出新算法具有对数据样本集更好的可扩展性.两种算法在领域半径 Eps 不同的情况下运行时间的变化.从图可以看出,在 DBSCAN 中,因为领域半径越大,进行区域查询的时间就越多,所以 Eps 增大,运行时

间也会增多.但是对于 DCUR 算法来说,Eps 增大,运行时间越短,这是因为随着 Eps 的增大,需要查询参考点的点越来越少,则运行时间就会相应的减少.

从两种算法在自行构造下的数据集的测试结果.可以看出,对于 DBSCAN 算法,Eps 小点,得出两个簇,稍微大点,得出三个簇,但是周围的一些噪声也被聚类到簇中,聚类的质量明显地降低.而对于 DCUR 算法,数据集可以更精确地被分为三个簇.测试结果表明 DCUR 算法能够提高聚类的质量,同时该算法表现出更好的动态适应性,能够处理些密度不均匀的数据来进行聚类分析.

5 结束语

本文是在算法 DBSCAN 的基础上,提出了一种使用 k 个参考点的快速密度的聚类算法 DCUR,DCUR 不仅保持了 DBSCAN 算法的优点,而且降低了聚类的复杂度,能够适用于大规模、高维、不均匀数据进行数据挖掘,试验和理论都证明了上述结论.我们下一步的工作是研究怎么样能够降低两个参数 Eps、MinPts 对基于密度的聚类产生的影响,减小算法对参数的依赖性,进一步提高基于密度聚类算法的精确性.

参考文献:

- [1] Kaufman L, Rousseeuw P J. Finding groups in data: an Introduction to cluster analysis [M]. New York: John Wiley and Son, NY, 1990:145-193.
- [2] Ng R, Han J. Efficient and effective clustering methods for spatial data mining [C]// Proc. of the 20th Conference on VLDB. Santiago, Chile, 1994:144-155.
- [3] Guhas Rastogir, Shim K. Cure: an efficient data clustering method for very large databases [C]// Proceedings of the ACM SIGMOD international conference on Management of data. New York, ACM Press, 1996: 103-114.
- [4] Ester M, Kriegel H P, Sander J, et al. A density based algorithm for discovering clusters in large spatial databases with noise [C]// Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining. Portland: AAAI Press, 1996: 226-231.
- [5] 曾依灵,许洪波,白硕.改进的 OPTICS 算法以及其在文本聚类中的应用[J].中文信息学报,2008,22(1): 51-55.

参考文献:

[1] 刘琳,于海斌. 异构无线传感器网络中簇首的优化部署策略[J]. 通信学报,2010,31(10):229-237.

[2] 李亚男,徐夫田,陈金鑫. 基于 LEACH 的 WSNs 分簇优化策略[J]. 传感技术学报,2014,27(5):670-674.

[3] 李建洲,王海涛,陶安. 一种能耗均衡的 WSN 分簇路由协议[J]. 传感技术学报,2013,26(3):396-401.

[4] 龚本灿,李腊元,蒋廷,等. 基于生成树的无线传感器网络分布式路由协议[J]. 微电子学与计算机,2008,25(11):77-80.

[5] 张明才,薛安荣,王伟. 基于最小生成树的非均匀分簇路由算法[J]. 计算机应用,2012,32(3):787-790.

[6] 陆晶,马悦,吴晓军. 一种基于最小生成树的非均匀分簇路由算法[J]. 计算机应用,2012,33(10):2293-2296.

[7] 朱敏,肖震,刘昊霖,等. WSN 中基于虚拟网格的分簇路由算法[J]. 四川大学学报(工程科学版),2012,44(5):143-148.

[8] 刘琳,于海斌. 异构无线传感器网络中簇首的优化部署策略[J]. 通信学报,2010,31(10):229-237.

[9] 翟春杰,徐建闽,刘永桂. 基于分区的能耗均衡路由协议[J]. 传感技术学报,2016,29(1):80-86.

[10] Smaragdakis G, Matta I, Bestavros A. SEP: a stable election protocol for clustered heterogeneous wireless sensor networks[C]// Proceedings of the 2nd International Workshops on Sensor and Actor Network Protocols and Applications. Boston, USA: IEEE, 2004: 223-233.

[11] 卢建刚,乐红兵. 基于区域划分的 WSN 非均匀分簇算法[J]. 计算机工程与设计,2011,32(8):2639-2642.

[12] 刘帅,李正炜,吴元昊,等. 基于能耗梯度的无线传感器网络路由算法[J]. 传感技术学报,2016,29(8):1247-1252.

[13] 张明才,薛安荣,王伟. 基于最小生成树的非均匀分簇路由算法[J]. 计算机应用,2012,32(3):787-790.

[14] Akyildiz I. Wireless sensor networks: a survey[J]. Computer Networks,2002,38(4):393-422.

作者简介:

李东林 男,(1992-),硕士研究生. 研究方向为无线传感器网络. E-mail: 540412133@qq. com.

韦素媛 女,(1971-),副教授,硕士生导师. 研究方向为计算机网络.

(上接第 35 页)

[6] 管涛. 统计聚类模型研究综述[J]. 计算机科学,2012,39(7):18-24.

[7] 郝晓丽,张靖. 基于改进自适应聚类算法的 RBF 神经网络分类器设计与实现[J]. 计算机科学,2014,41(6):260-263.

[8] 周开乐,杨善林,丁帅,等. 聚类有效性研究综述[J]. 系统工程理论与实践,2014,34(9):2417-2431.

[9] 赵杰,杨柳. 聚类分析算法 dBscan 的改进与实现[J]. 微电子学与计算机,2009,26(11):190-192.

[10] 许虎,王治和. 一种改进的基于密度的聚类算法[J]. 微电子学与计算机,2012,29(2):44-53.

作者简介:

闫安 男,(1977-),博士,副教授. 研究方向为市场营销.

刘琪林(通讯作者) 男,(1992-),硕士研究生. 研究方向为数据挖掘. E-mail:1635813128@qq. com.