第 37 卷 第 19 期
Vol.37 No.19

计 算 机 工 程
Computer Engineering

2011 年 10 月
October 2011

• 软件技术与数据库 •　　文章编号：1000—3428(2011)19—0065—03　　文献标识码：A　　中图分类号：TP301.6

空间数据聚类中的网格粒度求解方法

陈 曦，马一峰

(长沙理工大学计算机与通信工程学院，长沙 410114)

摘 要：提出一种空间数据聚类中的网格粒度求解方法。在网格动态划分过程中，根据密网格和稀疏网格的产生情况，确定最佳网格粒度与密度阈值。在给定一组密度阈值的条件下，利用该方法可以确定一个最佳的密度阈值及相应的网格粒度。给出该求解方法的聚类算法描述及算法时间复杂度分析。实验结果表明证明了该算法的有效性。

关键词：空间数据聚类；网格粒度；密度阈值；网格划分

Solving Method of Grid Granularity in Spatial Data Clustering

CHEN Xi, MA Yi-feng

(College of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha 410114, China)

【Abstract】This paper proposes a solving method of grid granularity in spatial data clustering. It ascertains the optimum grid granularity and density threshold on the basis of the situation in which dense and sparse grid is generated during the procedure of grid dynamical partitioning. Using an optimum density threshold and corresponding grid granularity can be ascertained with a set of given density thresholds. It gives the algorithm based on the proposed method and the time complexity analysis. Experimental results show the validity of the algorithm.

【Key words】spatial data clustering; grid granularity; density threshold; grid partition

DOI: 10.3969/j.issn.1000-3428.2011.19.020

1 概述

在空间数据聚类中，基于密度的聚类算法和基于网格的聚类算法是 2 种主要应用的算法。DBSCAN 算法[1]是典型的基于密度的聚类算法，其缺点之一是参数邻域半径 *Eps* 和密度阈值 *MinPts* 难以确定。DBSCAN 算法对多个数据集的实验都将 *MinPts* 值设为 4，然后根据 4-dist 图确定 *Eps* 值。依据 *k*-dist 图确定 *Eps* 值，在很大程度上依靠人的主观因素。在很多情况下，当 *MinPts* 设定为 4 时，无论怎样调整 *Eps* 值，都不可能得到理想的聚类结果。文献[2]针对一个有 3 个聚类的数据集进行实验，结果表明，该数据集仅在参数 *Eps*=2 且 *MinPts*=4 的组合下，才能得到正确的聚类结果。可见，邻域半径和密度阈值的取值均对聚类质量有重要影响。

网格和密度相结合的聚类算法综合了两者的优点，能发现任意形状的高密度类簇，同时又有较高的效率。如 DBSCAN 算法中的 *Eps* 和 *MinPts* 参数一样，这类算法中的网格粒度和密度阈值对聚类质量有决定性影响。

本文提出一种空间数据聚类中的网格粒度求法，依据密网格最多原则和数据点在网格中的均匀分布原则，根据密网格和稀疏网格的产生情况确定最佳网格粒度。

2 基于网格和密度的聚类算法

基于网格和密度的聚类算法的思想是：对数据空间进行网格化划分，将连通的密网格中的数据点划分到一个类中。算法涉及 2 个参数：每一维的等分数 *H* 和密度阈值 *MinPts*。

一些网格聚类算法中对网格的划分方法如下(*N* 表示数据点总个数)：文献[3]令 $H=\sqrt{N/coefM}$，*coefM* 是一个正整数的调节系数，由用户提供；文献[4]取 $H=\sqrt{N}$；文献[5]认为，在一般情况下 $H=\mathrm{int}(\sqrt{N})+1$ 是一个理想取值；文献[6]提出了在二分网格过程中对网格进行密度聚类的方法，该算法逐级二分每个网格成为等体积的 2 个部分，当一个网格的密度小于阈值则停止细分该网格。在上述文献中，文献[6]在数据空间的动态划分中根据数据点的密度进行网格划分，但没有解决最佳网格粒度问题，其他文献中的网格划分方法只考虑数据点总数一个因素，不能反映数据集的内部结构。

3 本文方法

3.1 一个密度阈值条件下的最佳网格粒度求法

把 *D* 维数据空间的每一维分为 *H* 等份，可以将数据空间分为 D^H 个大小相同的网格(为描述方便，下文均指二维空间)。若网格中的数据点个数大于或等于密度阈值，称该网格为密网格，若网格中的数据点数小于密度阈值而大于 0，则称该网格为稀疏网格，不包含数据点的网格称为空白网格。

对于给定的密度阈值 *MinPts*，好的网格划分应该使得跨越高密度区域和低密度区域的网格尽量少。为此，网格必须适当地小，使尽可能多的密网格密度接近 *MinPts*。基于这个事实，可以认为在给定 *MinPts* 值下，当密网格个数达到最大值时，网格粒度为最佳。

网格划分方法有 2 种：(1)从粗到细的方法，先把整个数据空间作为一个网格，然后将每个网格在每一维上都二等分，直到发现密网格个数减少为止，该方法找到的首个密网格极大值通常就是全局密网格最大个数。(2)从细到粗的方法，先确定一个细的网格粒度，然后将相邻的 4 个网格凝聚为一个网格，直到合并为一个网格为止。设数据点总数为 *N*，实际上，从粗到细的划分法可设定起始每维划分段数为：

$$m\_start = floor(\sqrt{N/MinPts})$$

其中，*floor* 为向下取整函数；*N* 为数据点总数；*m_start* 为

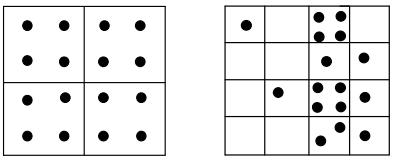
基金项目：国家自然科学基金资助项目(60973113)

作者简介：陈 曦(1963－)，男，教授，主研方向：数据挖掘，人工智能；马一峰，硕士研究生

收稿日期：2011-03-22　**E-mail：**chenxi_csu@yahoo.com.cn

假定二维空间中数据点完全均匀分布时为得到最多密网格而需要将每维划分成的段数。从细到粗的合并方法也可以进行到每维段数不大于 *m_start* 时结束。

2 种方法都可以保证得到密网格大致最多的那轮划分。其依据是：设在给定的二维数据空间中数据点是完全均匀分布的，则在给定 *MinPts* 条件下密网格个数达到最大值，即 $N/MinPts$ 个，同时网格达到最大尺寸(见图 1(a))；而非完全均匀分布情况下密网格则少于 $N/MinPts$ 个，且必须用更小的网格尺寸才能获得最多密网格(见图 1(b))。



(a)4 个密网格 (b)2 个密网格

图 1 数据点分布的密网格示例

3.2 一组密度阈值条件下的最佳密度阈值及网格粒度求法

根据给定的一个 *MinPts*，记该密度阈值下密网格最多的那轮划分为 *G_*最佳；记比 *G_*最佳细一级的网格划分为 *G_*细；记比 *G_*最佳粗一级的网格划分为 *G_*粗。有时用户很难给出一个好的密度阈值参数，但可以大致估计一个取值范围。设其中一个 *MinPts* 值是最好的密度阈值，此 *MinPts* 值应该略大于低密度区域的大多数局部密度极值(局部密度极值指包含一定数量点的一个局部高密度小区域的密度)。

因此，以下推断成立：(1)由于 *G_*最佳中大多数密网格的密度接近于 *MinPts*，因此 *G_*细中新产生的稀疏网格个数应该接近于 *G_*最佳中密网格个数的 4 倍；反之，若密网格的密度显著高于 *MinPts*，则由密网格细分所得的 4 个网格中，仍然存在密网格的可能性很大。(2)由于高密度区域的密网格基本相连，因此在由 *G_*最佳到 *G_*粗的合并过程中，最多的情况是 4 个密网格会合并成 1 个密网格。

满足上述要求的网格粒度既能较好地区分高密度与低密度区域，同时网格内数据点的分布又比较均匀，避免了在高密度区域内部形成过多稀疏网格，形成对数据点的一个较高层次上的压缩，如图 2 中网格 A 内数据点的分布较均匀。

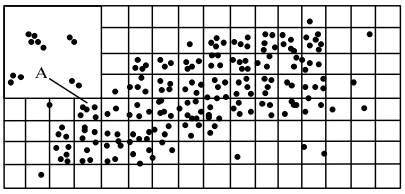


图 2 适当的网格粒度

考虑密度阈值过小或过大的情形：若 *MinPts* 过小，则网格也相当小，若小到小于数据的基本粒结构，会使得很多密网格中的数据点呈现偏倚的分布，即网格中数据点集中在某一局部，网格其他部分为空白。同时高密度区域中会存在一定的稀疏网格，而低密度区域也会出现一定密网格。如图 3 中网格 B 内数据点的分布是偏倚的。

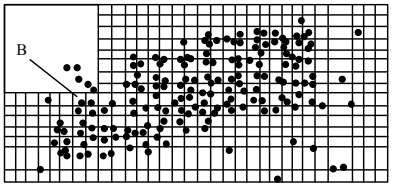


图 3 过小的网格粒度

若 *MinPts* 过大，则密网格的密度会存在显著差异，密度相对更高的密网格在细分过程中产生的稀疏网格个数会显著地小于 4。同时，相当部分密网格会一部分包含高密度区域的点，另一部分包含低密度区域的点，如图 4 中网格 C 即属于这种情况。

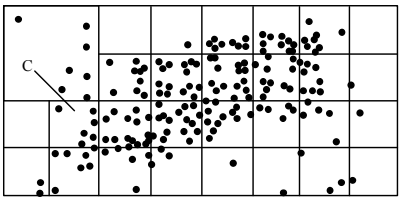


图 4 过大的网格粒度

在高密度区域内连续的密网格所占比例减少，所以在网格合并过程中由 4 个密网格合并成一个密网格的比例也低。记 *G_*最佳的密网格个数与 *G_*粗的密网格个数之比为 *INDEX_MERGE*，记 *G_*最佳细分过程中由密网格产生的稀疏网格个数与 *G_*最佳的密网格个数之比为 *INDEX_SPLIT*。根据上文分析，当网格处于某个适当大小阶段时，指标 *INDEX_MERGE* 接近最大值 4，当网格过小或过大时该指标均下降；指标 *INDEX_SPLIT* 也具有同样性质。由于这 2 个指标都是越大越好，因此本文取 *INDEX_MERGE* 与 *INDEX_SPLIT* 的积作为衡量网格粒度的最终指标，记为 *INDEX_MS*。

4 算法描述及时间复杂度分析

根据上述方法，本文提出一种聚类算法 BGDBSCAN。对网格进行由细到粗的合并，需要事先确定一个较细的网格大小。由于细网格编号决定了数据点在上一级粗网格中的唯一编号，因此不需要再针对数据点计算其所在的网格编号，大大减少了计算量。BGDBSCAN 算法描述如下：

输入 二维数据集 *D*、一组密度阈值 *MinPtses*、最小网格长度 *h_base*

输出 聚类结果 *S*

Step1 根据 *h_base* 将数据空间划分为细网络，并将数据点映射到相应网格中，采用哈希表作为网格和数据点的存储结构，网格坐标作为哈希表的 *Key* 值。

Step2 迭代地将矩形区域中的 4 个相邻网格合并为一个更粗的网格，直到每维段数不大于 *m_start* 为止，每轮合并的同时针对每个密度阈值统计密网格个数以及前一轮划分到本轮划分的密网格和稀疏网格的产生情况。

Step3 针对每个 *MinPts* 参数，选择密网格最多的那一轮划分为 *G_*最佳，比 *G_*最佳粗一级的划分为 *G_*粗、比 *G_*最佳细一级的划分为 *G_*细，根据 3 个网格划分计算 *INDEX_MS* 的值。

Setp4 选择 *INDEX_MS* 指标值最大的相应 *MinPts* 值作为最佳密度阈值参数，以该 *MinPts* 为基准的密网格最多的那轮划分中的网格为最佳网格。

Step5 按密度可达的原则对密网格进行聚类。

扫描数据点并将其存入相应细网格中的时间复杂度为 $O(N)$，N 为数据点总数。在网格合并过程中，对每个非空网格都需要针对各个 *MinPts* 值累计统计信息。设首轮网格划分中有 m 个细网格，每次合并中的绝大多数情况是将 4 个细网格合为 1 个粗网格，因此，多轮划分的总网格个数小于 $4m/3$，设共有 k 个 *MinPts* 值，则该过程的时间复杂度为 $O(4km/3)$。设 G_最佳中密网格的个数为 w，在将连通的密网格聚为一类的过程中，每个密网格存在 8 个相邻网格(边缘

区域少于 8 个)，所以该过程的时间复杂度为 $O(8w)$。因此，BGDBSCAN 算法的总时间复杂度为 $O(N+4km/3+8w)$，在大数据集情况下 $4km/3$ 或 $8w$ 均远小于 N。

5　实验结果及分析

本文采用 CLUTO 项目(见 http://glaros.dtc.umn.edu/gkhome/)中的 2 个数据集 t4.8k、t5.8k 进行实验，2 个数据集如图 5、图 6 所示。

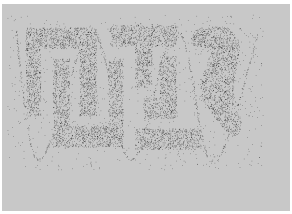


图 5　t4.8k 数据集

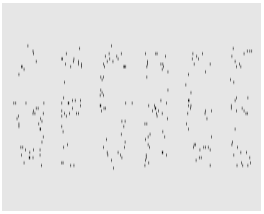


图 6　t5.8k 数据集

t4.8k 数据集和 t5.8k 数据集均包含 6 个主要聚类。实验结果如表 1、表 2 所示，其中仅列出为偶数的 *MinPts* 值。

表 1　t4.8k 数据集实验结果

MinPts	*INDEX_MS*	聚类个数
2	2.57	246
4	6.57	30
6	5.00	29
8	7.12	1
10	7.23	4
12	7.71	6
14	7.52	6
16	7.31	6
18	6.71	7
20	5.82	8

表 2　t5.8k 数据集实验结果

MinPts	*INDEX_MS*	聚类个数
2	3.63	301
4	4.13	34
6	7.26	9
8	7.61	6
10	7.02	10
12	5.80	14
14	4.12	22
16	7.04	6
18	7.40	6
20	7.53	6

可以看出，存在着较连续的 *MinPts* 取值区间，其 *INDEX_MS* 指标都是比较高的，针对该区间的 *MinPts* 值进行的聚类都得到了很好的结果。图 5 和图 6 中的主要类簇都得到正确的识别，没有因为细线的存在而造成不同类簇的融合，其原因是本文中求 *INDEX_MS* 的方法使得到的网格既足够地小，同时又不会过于陷入局部细节，因此，BGDBSCAN 算法能正确地分隔不同的类簇，同时又忽略了局部细节的影响，能够很好地发现主要的聚类结构。

6　结束语

快速而有效地得到合适的网格粒度和密度阈值，对基于网格和密度的聚类算法有重要意义。本文提出了一种网格粒度求解方法，通过分析网格细分(或合并)过程中数据点在网格中的分布情况，以及密网格和稀疏网格的产生情况，可以有效地确定较好的网格粒度和密度阈值。本文方法可以作为其他聚类算法(如基于抽样的聚类算法)的预处理步骤，也可直接利用该算法得到主要聚类结构。研究在完全无参数条件下的最佳网格粒度及密度阈值求法是下一步的方向。

参考文献

[1] Ester M, Kriegel H P, Sander J, et al. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise[C]//Proc. of the 2nd International Conference on Knowledge Discovering and Data Mining. Portland, USA: [s. n.], 1996.

[2] Halkidi M, Batistakis Y, Vazirgiannis M. Cluster Validity Methods[EB/OL]. [2011-01-15]. http://citeseer.ist.psu.edu/534869.html.

[3] Gao Song, Xia Ying. GDCIC: A Grid-based Density Confidence Interval Clustering Algorithm for Multi-density Dataset in Large Spatial Database[C]//Proc. of the 6th International Conference on Intelligent Systems Design and Applications. Washington D. C., USA: IEEE Computer Society, 2006.

[4] 周炎涛，易兴东，吴正国. 基于网格的带有参考参数的聚类算法[J]. 计算机工程, 2008, 34(9): 98-100.

[5] 薛丽香，邱保志. 基于密度可达的多密度聚类算法[J]. 计算机工程, 2009, 35(17): 66-68.

[6] 岳士弘，王正友. 二分网格聚类方法及有效性[J]. 计算机研究与发展, 2005, 42(9): 1505-1510.

编辑　顾姣健

(上接第 64 页)

4　结束语

本文利用 LDA 模型和极性词典实现了对餐馆评论的主题抽取和倾向性计算，结合用户给出的每个方面的得分和评论正文，用逻辑回归模型进行了训练。实验结果证明了本文方法的有效性。下一步可在大规模的语料上对 LDA 模型的参数进行训练，以达到更好的实验效果，在极性判定方面，可以结合其他的方法改进基于词典的方法，增加方法的灵活性。

参考文献

[1] 韩　丽，岑松祥，马　建，等. 基于博主之间社会关系的博客排序算法[J]. 计算机工程, 2010, 36(5): 52-53.

[2] 伍　星，何中市，黄永文. 基于弱监督学习的产品特征抽取[J]. 计算机工程, 2009, 35(13): 199-201.

[3] Zhu Jingbo, Wang Huizhen, Benjamin K, et al. Multi-aspect Opinion Polling from Textual Reviews[C]//Proc. of CIKM'09. Hong Kong, China: [s. n.], 2009.

[4] Lu Yue, Zhai Chengxiang, Sundaresan N. Rated Aspect Summarization of Short Comments[C]//Proc. of WWW'09. Madrid, Spain: [s. n.], 2009.

[5] 潘　宇，林鸿飞. 基于语义极性分析的餐馆评论挖掘[J]. 计算机工程, 2008, 34(17): 208-210.

[6] 徐琳宏，林鸿飞，潘　宇，等. 情感词汇本体的构造[J]. 情报学报, 2008, 27(2): 180-185.

[7] Blei D, Ng A Y, Jordan M I. Latent Dirichlet Allocation[J]. Journal of Machine Learning Search, 2003, 3(5): 993-1022.

编辑　顾姣健