一种基于代表点的增量聚类算法*

孟凡荣，李晓翠，周　勇
（中国矿业大学 计算机科学与技术学院，江苏 徐州 221116）

摘　要：针对现有的增量聚类算法对参数敏感度较高、时空复杂度较高等问题，提出了一种基于代表点的增量聚类算法。首先采用代表点聚类算法对静态的数据库进行聚类；然后根据新增加的节点与已存的代表点之间的关系，判断是否将其添加到已存的代表点所属的类簇中，或是提升为新的代表点；最后，再次采用代表点聚类算法对其进行聚类。实验结果证明，该算法对参数的敏感性低、效率高、占用空间小。

关键词：代表点；节点属性；增量聚类
中图分类号：TP311　　**文献标志码**：A　　**文章编号**：1001-3695(2012)08-2865-03
doi:10.3969/j.issn.1001-3695.2012.08.017

Incremental clustering algorithm based on representative points

MENG Fan-rong, LI Xiao-cui, ZHOU Yong
(*School of Computer Science & Technology, China University of Mining & Technology, Xuzhou Jiangsu* 221116, *China*)

Abstract: As the existing incremental clustering algorithms have various disadvantages such as high sensitivity to parameters, high time-space complexity, etc. This paper presented an incremental algorithm based on representative points. It first used the static clustering algorithm based on representative points to cluster the original data set. Then according the relationship between the new points and the existing representative points, the algorithm judged whether the new points should be added to the clusters containing the existing representative points or promoted as new representative points. Finally it used the static clustering algorithm again to cluster the new points. Experimental result shows that this algorithm is insensible to parameters, efficient and occupies little space.

Key words: representative points; properties of the new points; incremental clustering

0　引言

随着通信和网络技术的高速发展，信息量以前所未有的速度飞快增长。数据挖掘成为一个热门的研究方向，其核心课题就是聚类[1]。现有的静态聚类算法是采用某种算法将所有数据一次性分析，产生聚类结果[2~5]。但是随着信息量的不断更新，数据以流的方式不断增加，采用静态聚类算法需要重新对所有的数据进行分析聚类，时间复杂度较高，而且通常增量数据库的数据量非常大，占用空间较多。因此，增量聚类算法[6]的研究成为人们研究的新方向。当大型数据集的数据变化时，增量聚类算法仅对变化的部分数据增量地更新聚类结果，充分利用前次聚类结果，提高效率。增量聚类算法的另一个优点是不必将全部数据存储在内存中，对空间要求低。因此，增量聚类算法非常适合动态环境和非常大的数据集。

早期的增量聚类算法[7]是在 DBSCAN 算法的基础上进行改进的，该算法依次将新添加的数据与先前的数据进行比较，更新聚类结果。但是由于 DBSCAN 算法本身对于参数的敏感性较高、算法的鲁棒性较差、时间复杂度较高，而且算法在增量聚类过程中没有考虑新添加的数据之间的关系，聚类结果的准确性不高。陈宁等人[8]提出基于密度的网格聚类算法 GDCA，以发现大规模空间数据库中任意形状的聚类，大大降低了时间复杂度，但是对于空间的利用率仍然较低。

本文提出了一种基于代表点的增量聚类算法，从数据库中提取代表点仓库并进行聚类；对于新添加的数据，根据它们与已存的代表点之间的关系，判断是否将其添加到已存的代表点所属的类簇中，或是提升为新的代表点；最后采用代表点聚类算法对其进行聚类。该算法不需要重复地对整个数据库进行聚类，降低了时间和空间复杂度，对于参数的敏感度较低，算法的鲁棒性更好。

1　基于代表点的聚类简介

代表点（representative points）这一概念早在 2001 年就被陈恩红等人[9]提出用于聚类分析；2010 年，贾瑞玉等人[10]提出了基于代表点的快速聚类算法 REPBFC，采用簇中固定数量代表点来代表簇对象进行距离的计算，已达到快速聚类的效果。

2011 年，笔者提出了一种基于代表点的聚类算法。首先对数据库进行分析，采用寻找代表点算法得到一个能够代表整个数据库特征和数据分布特性的代表点仓库，然后通过对代表点仓库进行聚类，以展示数据库中不同类簇的形状；将其他能够被代表的节点进行存储，通过检测代表这些节点的代表点所属的类簇，将被代表的节点分配到相应的类簇。对于既不是代

收稿日期：2012-01-10；**修回日期**：2012-02-16　**基金项目**：国家教育部博士点基金资助项目(20100095110003)；国家博士后科学基金资助项目(20070421041)；江苏省博士后科学基金资助项目(0701045B)；中国矿业大学科技基金资助项目(2007B017)

作者简介：孟凡荣(1962-)，女，教授，博导，博士，主要研究方向为数据库技术、数据挖掘；李晓翠(1987-)，女，硕士研究生，主要研究方向为增量数据挖掘(xiaocuiworld@163.com)；周勇(1974-)，男，江苏徐州人，副教授，博士，主要研究方向为数据挖掘.

表点又不是被代表点的节点,将其存放到噪声节点中。以下给出代表点以及与本文算法相关的定义。

给定数据库 $V=\{v_1,\cdots,v_{|v|}\}$,类簇 c 为点的集合 $c=\{v_1,\cdots,v_{|c|}\}$,其中 $c\in V$,v_i 是一个 d 维的点 $v_i=\{v_{i1},\cdots,v_{id}\}$;类簇 C 是集合 $C=\{c_1,\cdots,c_{|C|}\}$。

定义 1 最近邻居。$NN(v_i)$ 为点 v_i 的邻居节点集,当 $\forall v_x\in V,\forall v_j\in NN(v_i),D(v_i,v_j)\leqslant D(v_i,v_x),v_j\neq v_x$,$D(v_i,v_j)$ 表示节点 v_i 与 v_j 的欧氏距离。

定义 2 代表点。r_i 为代表点,当 r_i 不能被代表点 $r_j(i\neq j)$ 代表,并且 r_i 的 D 邻域内存在大于等于 K 个邻居节点,称 r_i 为代表点,其 D 邻域内的邻居节点能被 r_i 代表。代表点集合 $R=\{r_1,\cdots,r_{|R|}\}$。

定义 3 能被代表的节点。若存在代表点集 $R=\{r_1,\cdots,r_{|R|}\}$,且 $R\neq\varnothing$,使得 v_i 在 r_j 的 D 邻域内,则称 v_i 为能被代表的点。能被代表的节点 v_i 被代表点 $r_k\in R$ 代表,$\forall r_x\in R$,$D(v_i,r_k)\leqslant D(v_i,r_x),r_k\neq r_x$。

定义 4 噪声。节点 v_i 为噪声,当节点 v_i 既不是代表点,又不能被任何一个代表点代表。噪声不属于任何一个类簇。

定义 5 相关密度。代表点 $r_i\in R$ 的相关密度定义为 $RD(r_i)$。

$$RD(r_i)=\frac{1}{|NN_D(r_i)|}\sum_{j=1}^{|NN_D(r_i)|}D(r_i,NN_D(r_i,j)) \tag{1}$$

其中:$NN_D(r_i)$ 是指 r_i 的 D 邻域内的邻居,$NN_D(r_i,j)$ 是指 r_i 的 D 邻域内的第 j 个邻居。

定义 6 密度相关。给定一个密度标准 α,当两个代表点 r_i 和 r_j 出现下列关系为真时为密度相关:

$$\begin{aligned}D(r_i,r_j)&\leqslant RD(r_i)\cdot\alpha\\D(r_i,r_j)&\leqslant RD(r_j)\cdot\alpha\end{aligned} \tag{2}$$

定义 7 $r_j\in c_i$ 且 $r_k\in c_i,r_j,r_k\in R$。当且仅当 r_j 和 r_k 满足定义 6 所定义的密度相关。

定义 8 能被代表的节点 $v_j\in c_i$,当 v_j 能被代表点 r_j 所代表,且 $r_j\in c_i$。

2 基于代表点的增量聚类算法

2.1 算法描述

当有一批新的数据集 $N(N_1,N_2,\cdots,N_n)$ 引入时,如果 $N_x\in N$ 能够被已存在的代表点所代表,即 N_x 的 D 邻域内存在有代表点 R_i,那么将 N_x 归类到代表点 R_i 所在的类簇中,同时修改代表点 R_i 的相关密度,这个过程可能会导致类簇的合并或是分裂。当代表点的相关密度增大时,可能导致类簇的合并。如图 1 所示,图 1(a)显示初始数据集具有 3 个类簇 6 个代表点,新增加的节点 $N_1\sim N_4$ 在类簇 3 中代表点 R_6 的 D 邻域中,故 $N_1\sim N_4$ 能够被代表点 R_6 所代表;(b)显示 R_6 的相关密度增大,原来与其密度不相关的代表点 R_5 现在与其密度相关,导致了类簇 2 与类簇 3 的合并。当代表点的相关密度减小时,可能导致类簇的分裂。如图 2 所示,图 2(a)显示初始数据集具有 3 个类簇 6 个代表点,新增加的节点 $N_1\sim N_4$ 在类簇 2 中代表点 R_4 的 D 邻域中,故 $N_1\sim N_4$ 能够被代表点 R_4 所代表;(b)显示 R_4 的相关密度减小,原来与其密度相关的代表点 R_5 现在与其密度不相关,导致了类簇 2 的分裂,新增类簇 4。

当新的节点 N_x 不能被已存在的代表点所代表时,那么将该节点与先前的噪声集合合并在一起,从中选取新的代表点并对该噪声集合进行聚类,并判断其他节点能否被新选定的代表点所代表。若能,判定其为非代表点,同时将其划分到代表它的代表点所属的类簇;否则,判定其为噪声。如图 3 所示,新引入的节点不能被已有的代表点所代表。故将它们与原始的噪声集合合并,根据定义 2 从中选出新的代表点 R_7,图 3(a)中由于 R_7 不与其他代表点密度相关,故产生了一个新的类簇 4,同时将 R_7 的 D 邻域内的节点划分到类簇 4 中,R_7 的 D 邻域外的节点划分到噪声集合中;(b)中由于 R_7 同时与类簇 1 中的代表点 R_3 和类簇 2 中的代表点 R_4 密度相关,导致了类簇 1 与类簇 2 的合并,同时将 R_7 的 D 邻域内的节点划分到该合并的类簇中,R_7 的 D 邻域外的节点划分到噪声集合中。

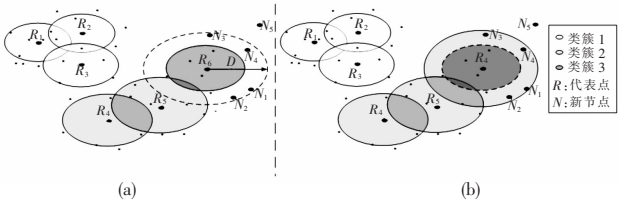

图 1 新引入的数据集导致类簇的合并

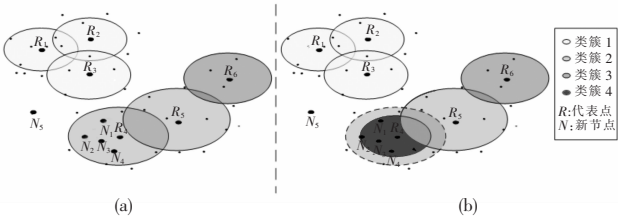

图 2 新引入的数据集导致类簇的分裂

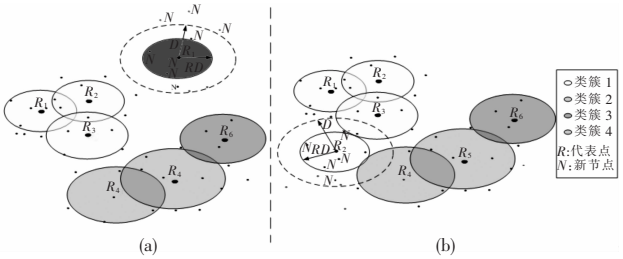

图 3 新引入的数据点提升为代表点后进行聚类

2.2 算法流程

首先判断新插入的节点的属性,有三种情况:a)能够被已存在的代表点所代表,成功聚类;b)提升为代表点,为下一步的聚类做准备;c)非代表点,可能被新提升的代表点所代表或者为噪声节点。

算法 1 判断节点属性算法

```
1  newRepresentive = ∅;
2  foreach N_x ∈ N
3    foreach R_i ∈ R
4      if( distance( N_x , R_i ) ≤D)
5      N_x . clusterNum = R_i . clusterNum;
6      beRepresented( N_x ) = true
7      increase_RD( R_i ) = true( or decrease_RD( R_i ) = true);
8      break;
9    if( beRepresented( N_x ) == true)
10     break;
11 else if ( Num_of_D( N_x ) ≥K)
12   createRepresentives += N_x ;
13   newRepresentive( N_x ) += N_x ;
14   isRepresentive( N_x ) = true;
15   increase_RD( R_i ) = true( or decrease_RD( R_i ) = true);
16 else isRepresentive( N_x ) = false;
```

当新插入的节点能够被已存在的代表点所代表,导致代表点的相关密度发生了变化,需要进一步判断该变化是否引起了类簇的合并或是分裂。

算法 2　类簇的合并或分裂算法

```
1   foreach R_i ∈ R
2     if( increase_RD( R_i ) == true )
3       foreach R_j ∈ R
4         if( R_i. clusterNum ! = R_j. clusterNum &&
5         distance( R_i , R_j ) ≤ v_i. clusterNum · α &&
6         distance( R_i , R_j ) ≤ v_i. clusterNum · α )
7         merge( Cluster_of_R_i , cluster_of_ R_j ) ;
8     if( decrease_RD( R_i ) == true )
9     foreach R_j ∈ R
10      if( R_i. clusterNum == R_j. clusterNum &&
11      ( distance( R_i , R_j ) > v_i. clusterNum · α ||
12      distance( R_i , R_j ) > v_i. clusterNum · α ) )
13      split( Cluster_of_ R_j ) ;
```

当新插入的节点不能够被已存在的代表点所代表,则通过代表点选取算法选取新的代表点,同时再次利用基于代表点的聚类算法对新的代表点进行聚类,并需要判断新产生的类簇是否引起了类簇的合并或是分裂。

3　实验分析

为了验证该算法对增量数据库聚类的有效性,并且将聚类分析结果可视化,本文采用如图 4 所示的数据集作为初始数据,可以明显看出,数据分布在三个区域内,采用基于代表点的聚类算法对该数据集进行聚类,其结果如图 5 所示。

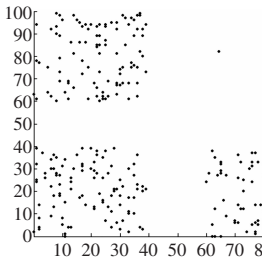

图 4　初始数据分布

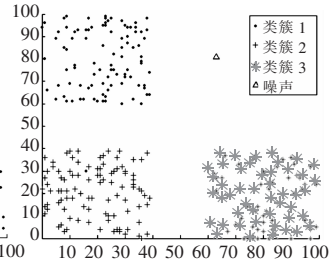

图 5　初始数据聚类结果

当有新的数据到达时,会导致先前聚类结果的变化。如图 6 所示,新到达的部分数据能够被已存在的类簇中的代表点所代表,故被划分到已有的类簇中,右上角的部分数据不能被其他类簇中的代表点所代表,故产生了新的代表点,并且这些代表点不与其他类簇中的代表点相互连接,导致了新的类簇的产生;另外的部分节点所产生的代表点与其他类簇中的代表点相互连接,或是这些节点的到达导致先前的代表点的相关密度发生变化,使得它们的相互连接关系发生变化,则可能会引起类簇的合并或是分裂。如图 7 所示,新到达的部分节点导致了先前的类簇 1 与类簇 2 的合并,其他的既不是代表点也不能被其他的代表点所代表,将其归类为噪声节点。

实验结果充分证明了该算法的可行性,并且由于该算法充分利用了先前的聚类结果,对于新到达的数据集,判断节点的属性为能够被代表的节点或是提升为代表点,对于提升为代表点的新节点,只需要判断其 D 邻域的代表点与它的相互连接关系,降低了算法的时间复杂度,同时由于利用了代表点这一特性,对于空间要求也大大降低。

为了验证该增量聚类这一过程,本文采用了有利于观察的、区域分布明显的初始数据集,但是在实际情况中,数据的分布和属性非常复杂,多次迭代之后,会导致误差累计增大。这也是增量聚类的一个重要特征之一,即用精度换取效率。

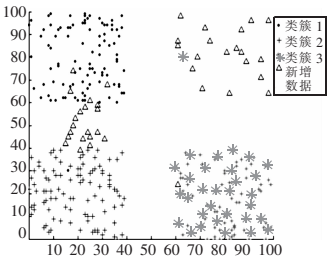

图 6　新的数据加入到数据集

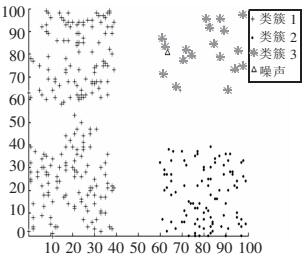

图 7　新数据的到达导致聚类结果发生变化

4　结束语

现有的增量聚类算法存在着诸多不足,如对参数敏感度较高、时空复杂度较高、算法的精度较低等问题。本文针对上述问题,提出了一种基于代表点的增量聚类算法,采用代表点聚类算法对于静态的数据库进行聚类,然后根据新增加的节点与已存在的代表点之间的关系,判断新到达的节点属性,判断是否将其添加到已存在的代表点所属的类簇中,或是提升为新的代表点,再次采用代表点聚类算法对其进行聚类。

实验结果证明,该算法效率高、占用空间小。但是由于该算法是在先前的聚类结果的基础上进行聚类,会导致误差像滚雪球一样,积累到一定程度后,误差将会超过最大容忍度。所以,如何将迭代次数和误差容忍度等问题考虑在内,将是下一步的主要研究工作。

参考文献:

[1] HAN Jia-wei, KAMBER M. 数据挖掘概念与技术[M]. 范明, 孟小峰, 等译. 北京:机械工业出版社, 2006.

[2] SONG Qin-bao, MARTIN S. Mining Web browsing patterns for e-commerce[J]. Computer in Industry, 2006, 57(7): 622-630.

[3] KIM K J, CHO S B. Personalized mining of Web documents using link structures and fuzzy concept networks[J]. Applied Soft Computing, 2007, 7(1): 398-410.

[4] WANG Chao, LU Jie, ZHANG Guang-quan. Mining key information of Web pages: a method and its application[J]. Expert Systems with Applications, 2007, 33(2): 425-433.

[5] 李石君, 于俊清, 欧伟杰. 基于 HTML 模式代数的 Web 信息提取方法[J]. 计算机研究与发展, 2006, 43(9): 1644-165.

[6] ESTER M, KRIEGEL H P, SANDER J, *et al*. Incremental clustering for mining in a data warehousing environment[C]//Proc of the 24th International Conference on Very Large Data Bases. San Francisco: Morgan Kaufman Publisher, 1998.

[7] 刘建晔, 李芳. 一种基于密度的高性能增量聚类算法[J]. 计算机工程, 2006, 32(21): 76-78.

[8] 陈宁, 周龙骧, 陈安. 基于密度的增量式网格聚类算法[J]. 软件学报, 2002, 13(1): 1-7.

[9] 陈恩红, 王上飞, 宁岩, 等. 一种利用代表点的有效聚类算法设计与实现[J]. 模式识别与人工智能, 2001, 14(4): 417-422.

[10] 贾瑞玉, 耿锦威, 宁再早, 等. 基于代表点的快速聚类算法[J]. 计算机工程与应用, 2010, 46 (33): 121-123, 126.

[11] 王洪春, 彭宏. 基于模糊 C-均值的增量式聚类算法[J]. 微电子学与计算机, 2007, 24(6): 156-157, 161.