第 34 卷 第 23 期 计 算 机 工 程 2008 年 12 月
Vol.34 No.23 Computer Engineering December 2008
·软件技术与数据库· 文章编号：1000—3428(2008)23—0059—03 文献标识码：A 中图分类号：TP311

食品 HACCP 分类的 BIRCH 算法

陈绍彬，叶飞跃，刘佰强，金 涛
(上海大学计算机科学与工程学院，上海 200072)

摘 要：食品卫生的 HACCP 自动分类要处理的数据集形状呈现多样性，对分类结果的准确性和专业性要求很高，已有的算法难以满足。该文基于经典 BIRCH 算法，结合多阈值思想和多代表点特征树思想，提出多阈值多代表点的 BIRCH 算法，增加了专业分类知识的指导，并对每一个代表点设立单独的阈值，使得该算法能适应各种形状的数据集，减少了聚类特征树重建次数，提高了算法的效率。
关键词：BIRCH 算法；聚类特征树；多代表点；多阈值

BIRCH Algorithm of Food HACCP Classification

CHEN Shao-bin, YE Fei-yue, LIU Bai-qiang, JIN Tao
(Department of Computer Science and Engineering, Shanghai University, Shanghai 200072)

【Abstract】 The HACCP data of food shows diversity shapes, its classification results on the accuracy and professionalism. The existed algorithms have been difficult to meet it. Based on the classic BIRCH algorithm, and the existed two algorithms multi-threshold and multi-representation points CF tree, a new multi-threshold and multi-representation BIRCH algorithm is designed, and the professional knowledge of the classification is added to guide set different variable thresholds to every representation points. Thus, the new algorithm can meet diversity data shapes, reduce the times of reconstruction of the CF tree, and improve the efficiency of the algorithm.
【Key words】 BIRCH algorithm; cluster feature tree; multi-representation point; multi-threshold

危害分析和关键控制点(Hazard Analysis and Critical Control Point, HACCP)，是生产(加工)安全食品的一种控制手段。通过对原料、关键生产工序及影响产品安全的人为因素等进行分析，确定加工过程中的关键环节、过程控制方法对加工过程的每一步进行的监视和控制，降低了危害发生的概率。HACCP 分类的自动处理是国家“十一五”计划关于食品卫生方面的重要研究内容之一。每一个要处理的 HACCP 数据信息就像一张张包含了一定信息的卡片，HACCP 的自动分类是对这些抽取到的数据信息的分类，HACCP 聚类分析的数据集是自然形状的增量式数据源，对处理结果专业性和精确性有较高的要求。

1 问题的提出

1.1 聚类分析的基本概念

聚类分析是数据挖掘中的重要组成部分，聚类分析依据的原则是使同一聚簇中的对象具有尽可能大的相似性，而不同聚簇中的对象具有尽可能大的相异性。本文重点研究层次聚类算法。层次聚类的经典算法有CURE[1]算法和BIRCH[2]算法，借鉴CURE算法的代表点思想，采用BIRCH算法的基本思路来对HACCP信息进行处理。

BIRCH 算法是一种支持增量式数据源的基于距离的聚类算法，由 T.Zhang 等提出，该算法有 2 个概念：聚类特征(Clustering Feature, CF)和聚类特征树(CF-tree)，通过这 2 个概念对簇进行概括，利用各个簇之间的距离，采用层次方法的平衡迭代对数据集进行归约和聚类。

1.2 BIRCH 算法的缺陷

BIRCH 算法具有如下缺陷：

(1)单一阈值。BIRCH 算法采用统一阈值形成多个簇，当 CF-Tree 无法处理所有数据时，即需要确定新的阈值，然后对 CF-Tree 重建，新的阈值一旦确定下来就保持不变，如此循环后最终的处理结果只是比较适合体积相差不大的簇之间的聚类。

(2)单一代表点。在阶段 1 中聚类特征树是从一个簇中只获得一个代表点，在聚类特征树中，每个节点只有一个代表点作为聚类特征。在阶段 2 中利用阶段 1 产生的聚类特征进行聚类，单一的聚类特征由于阈值的限制不仅只能包含有限个节点，而且使得算法只适合于球形数据集，从而无法胜任自然数据形状集合的处理。

针对以上问题，提出多阈值多代表点的 BIRCH 算法。

2 多阈值多代表点特征(M-MRF)树

2.1 代表点

定义 1(代表点)[3] 从一个簇中选择出的在形状上具有一定代表性的n个点，通过收缩因子s，将这些点向质心收缩所得到的点称为簇的代表点。

M-MRF-Tree 中的代表点有 2 种来源：(1)由聚类数据簇中根据距离抽取获得；(2)为了增加必要的聚类指导，根据已有的成功聚类经验和专业术语等特征封装成的代表点。

2.2 多代表点特征

定义 2(多代表点特征)[3] 将每个代表点的信息规范描述成聚类特征，一个节点具有多个代表点，这个节点的聚类特征就由多个聚类特征共同描述，称为多代表点特征。

基金项目：上海大学研究生创新基金资助项目“食品安全法规、标准文献信息平台建设——乳制品项目”(A.16-0108-07-001)
作者简介：陈绍彬(1977－)，男，硕士研究生，主研方向：数据挖掘，语义技术，信息抽取；叶飞跃，教授；刘百强、金 涛，硕士研究生
收稿日期：2008-04-14 **E-mail：**shaobin.chens@gmail.com

设簇中有 n 个具有 d 个属性的数据对象，即{ $\boldsymbol{o}_i$ }。其中，$\boldsymbol{o}_i=\{o_{i1},o_{i2},\cdots,o_{id}\}$，$i$=1,2,$\cdots$,$n$，定义为一个四元的数组如下：

$$M-MRF=(n,\boldsymbol{LS},SS,Rep(N,O,T))$$

其中，$\boldsymbol{LS}$ 是 n 个对象的线性和($\sum_i^n \boldsymbol{o}_i$)；$SS$ 代表平方和，即 $\sum_{i=1}^{n}\boldsymbol{X}_i^2$ ；$Rep(N,(O,T)_i)$ 是簇的代表点集(可变数组)，用来代表这个簇；N 为代表点的个数；O 是代表点；T 是对应于该代表点的阈值。簇之间的距离采用欧几里德距离公式得到。

2.3 可变多阈值

可变多阈值思想是每个代表点对应一个可变的阈值，节点的阈值通过其包含的代表点的阈值动态计算而来，但 $T_{\max}$ 是不变的。如此，既满足了 HACCP 聚类对可变阈值的需要，也防止了聚类结果数据爆炸的情况。

2.4 多阈值多代表点特征树(M-MRF-Tree)

M-MRF-Tree 是一棵高度平衡树，存储了层次聚类的簇的统计特征与几何(物理)特征，内部节点的分支因子为 B，叶子节点的分支因子为 L，每个叶子节点有 2 个指针，分别指向左右两边的叶子节点，这些指针将 CF 树中所有叶子节点链接起来以提高查询速度；代表点的阈值为 T_i ，节点的阈值为 T_{node}。分支因子 B 限定了每个内部节点最多含有的孩子的个数，阈值 T_i 是指与距离最近的代表点的距离，它限定了存放在叶子节点中簇的大小。

代表点随着簇的边界和形状的变化而变化，算法所能处理的簇的形状也动态地发生变化，使得聚类特征树适合各种数据形状，提高了数据边界的处理能力。

3 多阈值多代表点 BIRCH 算法说明

多阈值多代表点 BIRCH 算法(M-MRF-BIRCH)分下面 3 个阶段：

(1)M-MRF-BIRCH 扫描初始数据集，选择代表点构造一棵 M-MRF 树。

(2)将封装着专业指导信息的特征信息作为代表点插入到相应的节点中。

(3)用多阈值多代表点的 M-MRF 聚类特征树进行聚类分析。

3.1 多阈值多代表点算法

3.1.1 非代表点的选择算法[3]

算法步骤如下：

(1)定义代表点的数目。

(2)对数据簇求质心。

(3)根据欧几里德距离公式求得距离质心距离最远的点作为第 1 个代表点。

(4)从第 2 个代表点起依次是距离第 1 个代表点最远的点。

3.1.2 阈值的确定

根据数据集和内存情况确定特征树的阈值 $T_{\max}$ 和一个初始质心的阈值 T_0，代表点与质心的距离定义为该代表点的阈值 T_i ，$T_0\le T_i\le\frac{1}{2}T_{\max}$。

假如一个节点有 m 个代表点，则阈值为

$$T_{node}=2\times\max\{T_i,i=0,1,\cdots,m\}$$

3.2 M-MRF 树的操作

聚类特征树的操作主要是数据对象的插入操作，分成新数据点的插入和重建特征树时旧的数据重新插入新树的操作。插入数据对象的过程中需要计算对象的距离，用作阈值比较，距离计算公式如下：

$$RDis_{(Data-Node)}=\min\{Dis_{(Data-Rep)1},Dis_{(Data-Rep)2},\cdots,Dis_{(Data-Rep)m}\} \quad (1)$$

3.2.1 新数据点的插入

从根节点开始，由上而下查找与该插入数据距离最近的节点，达到叶子节点后，查找距离该新数据点最近的代表点，测试阈值 T_i ， T_{node} ，若阈值条件满足，则更新 CF 值；若 T_i 不满足而 T_{node} 满足，则插入一个新的条目到叶子节点，若插入后这个节点的分支因子不超过 L 则插入过程完成，更新 CF 值，否则要进行分裂操作。插入新的数据点之后，要对路径上的每个内部节点的 M-MRF 信息进行更新，并重新计算代表点。

若要执行分裂操作，则插入新的聚类特征到被插入点的父节点来说明新节点的出现。若父节点有空间存放该数据对象信息，只需要更新高一层父节点的节点信息以反映更新；否则要对父节点进行分裂，与上面分裂过程相同。具体算法如下：

算法 1 新数据点的插入

```
输入 HACCP 数据对象
输出 M-MRF-Tree
Node insertIntoM_MRFTree (Tree t, Node currentNode, Entry newEntry ){
  if (当前节点是叶子节点 ){
    找到了最近的代表点
      if(有代表点的阈值能够满足)
        吸收并重新计算代表点
      else{
          插入新条目
          if(条目数目小于分支因子 L)
              吸收并重新计算代表点
          else
              分裂新的节点
          }
   }
  else{
      查找最近的节点
      递归插入新数据对象
      if(不需要分裂)
          更新 M_MRFTree
      else{
          更新聚类特征;
          分裂节点
          if(不需要再分裂)
              合并当前节点
          else{
              if(当前节点 = = 根节点)
                  成功返回
              else
                  递归返回
              }
          }
      }
}
```

3.2.2 M-MRF 树的重建

由于内存的限制，当一棵 M-MRF 树增长到一定的高度，需要对 M-MRF 树进行重建。首先增大树的阈值 $T_{\max}$ 以降低树的高度，设原有的树的阈值为 T_{oldT} ，新树的阈值为 T_{newT} ，

代表点的阈值 T_i，$T_i \le \frac{1}{2}T_{newT}$。

重建过程如下所述：

(1)在新树中生成对应于旧树的当前路径。

(2)将旧树中当前路径中的节点插入到新树。对旧树当前路径中的每一个叶子节点，用代表点计算插入节点与新树中节点的距离，判断节点的插入位置。根据新树的阈值 T_{newT} 进行测试，满足 $Dist \le \frac{1}{2}T_{newT}$，按照算法 1 的方法进行操作。并重新计算代表点，更新节点的信息。

(3)释放新树和旧树中空闲的空间。旧树中一条路径上所有的叶子节点处理完毕后，把这条路径上所有的节点及叶子删除。新树中某些节点向前推移，释放空闲的位置。

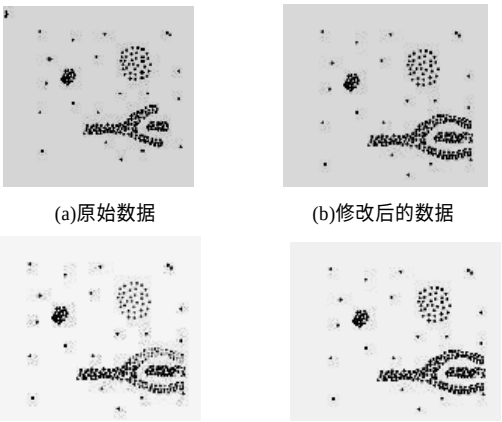
算法 2 M-MRF 树的重建

输入：新的阈值 T'_{max}，旧的 M-MRF 树(oldTree)
输出：新的 M-MRF 树(newTree)
初始化：一棵新树；T'_{max} = CONST
获得旧树中所有根到叶子节点的路径
for(对于每一条路径){
  拷贝到新树中
  for(路径中的每个节点)
    for(节点中的每个代表点)
      用算法 1 插入该代表点到新树
    重新选择代表点
  释放老树所占空间
}

4 实验及结果

4.1 算法处理数据集的能力

图 1(a)为采用著名算法 DBSCAN 的实验数据。为了充分展示本聚类算法多阈值在数据集边界处的处理能力，对该数据集做了相应的修改，将“Y”字部分的数据与其右边的数据形状变长，使得数据充分接近，如图 1(b)所示。经典 BIRCH 算法的处理结果如图 1(c)所示，由于其代表点数据的单一，它难以处理每一个数据簇，因此将“Y”字上面的部分数据归类到中间的“一”字形数据簇中，也因为阈值单一的原因，在数据边界较近的地方无法将数据准确地区分开来。图 1(d)所示是本文算法 M-MRF BIRCH 算法的聚类结果。



(a)原始数据　(b)修改后的数据

(c)经典 BIRCH 算法处理结果　(d)M-MRF BIRCH 算法聚类结果

图 1　2 种算法所能处理数据集的形状对比

4.2 算法的执行性能分析

由于 HACCP 分类是该食品卫生项目提出的新概念，该算法的实验数据目前只能借用其他该数据挖掘的实验数据集。本文算法使用的实验数据来源于 http://www.cs.waikato.ac.nz/ml/weka/index_datasets.html，该数据集共有 238 643 条数据记录，包括污染、学位、房屋等共 36 个数据种类。

本次测试是在 Windows XP 操作系统下用 Java 语言实现。测试环境是：处理机为 P4, CPU 2.66 GHz, 768 MB 的内存，硬盘 80 GB，数据库是 MySQL。对该数据集分别从 50 000, 100 000, 150 000, 200 000, 238 643 共 4 个不同的数据量处进行了 5 次聚类时间的测试，求得的平均数据量和聚类时间的序偶对(数据量；时间 sec(BIRCH)；时间 sec(M-MRF BIRCH))分别为{(50 000, 17.31, 10.51), (100 000, 31.42, 13.02), (150 000, 53.11, 16.43), (200 000, 60.54, 17.57), (238 643, 78.07, 19.10)}，如图 2 所示。

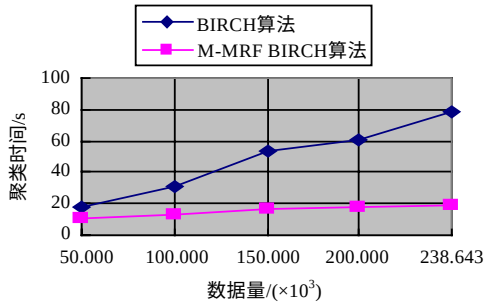


图 2　经典 BIRCH 算法和 M-MRF BIRCH 算法效率比较

由于多阈值多代表点的 BIRCH 算法 M-MRF 聚类特征树的在代表点的聚类思想和阈值的可变性，使得聚类特征树的重建次数比经典 BIRCH 算法大大减少。5 个阶段的效率平均提高 38.18%(50 000), 58.56%(100 000), 69.06%(150 000), 70.48%(200 000), 75.73%(238 643)，总体提高 62.40%。

5 结束语

本文提出的多阈值多代表点的 BIRCH 算法，将专业指导信息作为代表点加入到节点，将原来的一个节点一个阈值的策略改进为一个代表点一个可变阈值，节点阈值采用动态计算的方法，减少了聚类特征树的重建次数，增加了聚类的专业性，解决了单一聚类中心造成的聚类数据簇形状的限制和数据边界处理的不精确性，提高了聚类的执行效率和准确性。

参考文献

[1] Guha S, Rastogi R, Shim K. CURE: An Efficient Clustering Algorithm for Large Databases[C]//Proc. of ACM-SIGMOD Int'l Conf. on Management of Data. Seattle, Washington, USA: [s. n.], 1998: 73-84.

[2] Zhang T, Ramakrishnan R, Livny M. BIRCH: An Efficient Data Clustering Method for Very Large Databases[C]//Proc. of the International Conference Management of Data. Montreal, Canada: [s. n.], 1996: 103-114.

[3] 黄添强，秦小麟，王金栋. 多代表点特征树与空间聚类算法[J]. 计算机科学, 2006, 33(12): 189-195.