文章编号:1001 - 9081(2006)02 - 0379 - 04

基于特征向量的分布式聚类算法

李锁花,孙志挥,周晓云
(东南大学 计算机科学与工程系, 江苏 南京 210096)
(li_suohua@ hotmail. com)

摘 要:提出了一种新的表达数据集的方法——特征向量,它通过坐标和密度描述了某一密集空间,以较少的数据量反映站点数据的分布特性。在此基础上提出了一种基于特征向量的分布式聚类算法——DCBFV(Distributed Clustering Based on Feature Vector),该算法可有效降低网络通信量,能够对任意形状分布的数据进行聚类,提高了分布式聚类的时空效率和性能。理论分析和实验结果表明 DCBFV 是高效可行的。

关键词:数据挖掘;分布式聚类;特征向量
中图分类号: TP311 **文献标识码**:A

Distributed clustering algorithm based on feature vector

LI Suo-hua, SUN Zhi-hui, ZHOU Xiao-yun
(*Department of Computer Science and Engineering, Southeast University, Nanjing Jiangsu* 210096, *China*)

Abstract: Distributed clustering is a new research field of data mining. A new idea to sketch site data called feature vector was proposed, which described a dense space through coordinate and density, so captured distribution characteristic of data efficiently. Then a novel approach named DCBFV (distributed clustering based on feature vector) based on feature vector was proposed. DCBFV can decrease network overload, discover clusters with arbitrary shape, and improve the quality of global clustering effectively. Both theory analysis and experimental results confirm that DCBFV is feasible and effective.

Key words: data mining; distributed clustering; feature vector

0 引言

分布式数据挖掘是数据挖掘技术和分布式计算的有机结合,用于分布式环境下数据模式的发现。考虑到现有网络的带宽和安全性,把站点的全部数据集中到某一个中心站点聚类几乎是不可能的。首先站点间全部数据在网络间的传输会占用太多的网络资源,其次所有站点数据集中在一起,数据量会非常庞大,聚类效率会显著降低,传统的单机版的聚类算法[1~4]已经不适应分布式环境下的聚类问题。因此,探索新的有效的分布式聚类算法是目前数据挖掘的研究热点。

E. Januzaj 等人相继提出了基于密度的分布式聚类方法 DBDC[5] 和 SDBDC[6],在两个层次执行聚类:局部聚类和全局聚类。先在各个站点进行聚类,选择能够反映数据分布特征的局部代表对象,然后发送到中心站点进行全局聚类,最后把聚类结果返回到各个站点,以便站点把数据点划分到全局聚类中。其中代表对象是原始数据的一个抽取,可减少传输的数据量,降低网络开销。在这种框架下,决定分布式聚类精度和效率的关键是局部代表对象以及局部聚类和全局聚类算法的选择。DBDC[5] 在每个站点采用 DBSCAN 算法聚类,得到核心对象的集合,对核心对象进行筛选,调整 值和该邻域内的数据集,得到局部代表对象,其中局部代表对象的属性是坐标值和 值。中心站点的全局聚类也采用 DBSCAN 算法。

DBDC 的算法效率不是很理想,首先数据量为 n 时,DBSCAN 算法的时间复杂性为 $O(n*n)$;在空间索引如 R * -树[7] 的支持下,其复杂性为 $O(n*\log n)$,而建立索引通常需要消耗大量的时间;同时,筛选核心对象得到局部代表对象需要额外的操作,带来了计算复杂度的提高;而局部代表对象 ε 值和邻域内的数据集的改变使它不再符合核心对象的定义,这种表达方式带来了误差;在全局聚类时,只是根据局部代表对象本身的坐标属性进行聚类,并没有考虑到局部代表对象 ε 邻域内数据点的密集程度,从而失去了全局对局部数据的透视性,导致聚类质量的下降。SDBDC 算法通过计算各站点每个数据点的代表质量,按照逆序排列,选择代表性最强的点发送到中心站点,提高了全局聚类的精度。但是在站点上每选出一个代表点都需要迭代动态代表质量排序的算法,付出的时间代价是可观的。

本文在两层聚类结构的基础上,引入特征向量的概念,提出了一种基于特征向量的分布式聚类算法 DCBFV,在一定程度上弥补了上述算法在精度和效率上的不足。特征向量能够高效地反映数据分布特性,各站点可以在近似线性时间产生特征向量,且数据量远小于 DBSCAN 中产生的核心对象,因此可以作为局部代表对象传送到中心站点,有效降低了网络通信开销。在中心站点采用基于全局特征向量的聚类算法,充分考虑各站点数据的密集程度,提高了聚类的精度。

1 DCBFV 算法

1.1 相关概念与结论

定义 1 F 是 k 维空间中任意一点,满足以下两个条件时

收稿日期:2005 - 08 - 15;**修订日期**:2005 - 10 - 26 **基金项目**:国家自然科学基金资助项目(70371015)

作者简介:李锁花(1981 -),女,江苏丹阳人,硕士研究生,主要研究方向:数据挖掘与知识发现; 孙志挥(1941 -),男,江苏南通人,教授,博士生导师,主要研究方向:复杂系统集成、数据库、知识发现、数据挖掘; 周晓云(1979 -),男,江苏太仓人,博士研究生,主要研究方向:数据挖掘.

称为特征点：

1）以 F 为球心，半径为 ε 的空间内至少含有 $MinPts$ 个数据点。

2）该空间内所有数据点的重心与 F 相距小于等于 $a \times \varepsilon$。其中 a 为误差因子，这里建议取值为5% ~ 10%。重心的坐标为所有数据点在各维上的平均值。

定义2 以特征点为球心，半径为 ε 的 k 维空间称为特征空间，用 D 表示。位于同一特征空间的数据点属于同一聚类。

定义3 基于特征点 F 的 $k+2$ 维元组 $\boldsymbol{FV}(C,N,No)$ 称为特征向量，其中 C 为特征点 F 的 k 维坐标；N 为特征空间 D 内数据点的个数，即密度；No 为所属聚类的类标号。

特征点、特征空间、特征向量是一一对应的。在一定的误差范围内，特征点可以作为重心来代表密集区域的所有数据点，结合相应特征空间的密度，特征向量能够充分反映数据的空间几何分布特性，聚类标号是指特征空间内所有数据点所属的类标号，可以取0，1，2… 当 $FV.No=0$ 时，表示特征空间内所有数据点为噪声。

定义4 基于站点数据集的特征向量称站点特征向量（Site Feature Vector），用 $SFV(C,N,No)$ 表示；基于所有站点特征向量集的特征向量称全局特征向量（Global Feature Vector），用 $GFV(C,N,No)$ 表示。

定理1 给定两个 FV，当它们相应的特征点距离小于 2ε 时，这两个 FV 相应特征空间内的数据点属于同一聚类。

证明 设两个 FV 为 FV_1，FV_2，相应的特征点为 F_1，F_2，相应的特征空间为 D_1，D_2。当 $dist(F_1,F_2)<2\varepsilon$ 时，说明 D_1，D_2 有重叠部分，即 $D_1 \cap D_2 \neq \varnothing$，设 P 为这个重叠空间的一个数据点，即 $P \in D_1 \cap D_2$，那么 P 的类标号即等于 $FV_1.No$，又等于 $FV_2.No$。所以 $FV_1.No=FV_2.No$。说明这两个 FV 相应特征空间内的数据点属于同一聚类。

1.2 DCBFV 的框架

基于特征向量的分布式聚类算法（DCBFV）由三部分组成：

1）产生站点特征向量：在各个站点用 GSFV 算法产生站点特征向量集 $SFVset$。

2）全局聚类：在中心站点对所有站点特征向量进行聚类，并对它们标识聚类结果。

3）站点数据的分类：在各个站点扫描数据集，对每个数据点标识聚类结果。

1.3 产生站点特征向量

在站点用产生站点特征向量（Generate Site Feature Vector，GSFV）算法对数据进行处理，采用一定数目的特征向量来有效表示这个站点数据的分布特征，发送到中心站点进行聚类。GSFV 主要思路如下：

1）初始化候选站点特征向量 $CSFV(C,N,CS)$，其中 C 为候选特征点的 k 维坐标值，N 为候选特征空间的密度，CS 为 k 维元组，每一维的值是候选特征空间内所有点相应维的坐标算术和，$CSFV$ 存放在集合 $CSFVset$ 中。

2）扫描数据集，数据点 P 的坐标值用 k 维元组 C 表示，若它属于某一 $CSFV$ 相应的候选特征空间，重新计算 $CSFV$ 的属性值；若不属于任何 $CSFV$ 相应的候选特征空间，则产生一个 $CSFV$。

3）计算 $CSFV.C$ 与重心的距离，当距离大于 $a \times \varepsilon$ 则调整 $CSFV$ 的属性值，回到第2步继续执行，以使 $CSFV.C$ 逐步逼近重心，当距离小于等于 $a \times \varepsilon$，则认为 $CSFV.C$ 在误差范围内可以代表重心。转向4）。

4）比较 $CSFV.N$ 和密度阈值 $MinPts$，如果 $CSFV.N < MinPts$，说明 $CSFV$ 代表噪声区域，可以直接从 $CSFVset$ 中删除。如果 $CSFV.N \geqslant MinPts$，说明 $CSFV$ 代表密集区域，赋值给 $SFV(C,N,No)$ 的相应属性，得到站点特征向量的集合 $SFVset$，同时将 $CSFV$ 从 $CSFVset$ 中删除。

算法 GSFV

输入 $DataSet$：k 维数据集；ε：空间半径；$MinPts$：密度阈值；a：误差因子

输出 $SFVset$：站点特征向量集

步骤：

```
SFVset = ∅
// 初始化，随机选择一个点
CSFVset = {( P.C,0,0)}
while (CSFVset ≠ ∅){
  for (int i = 1; i < = DataSet.size; i++){
    for each CSFV with dist(Pi.C ,CSFV.C) < = ε
      { CSFV.CS = CSFV.CS + Pi.C;
      CSFV.N = CSFV.N + 1; }
    if no CSFV that dist(Pi.C ,CSFV.C) < = ε
      /* Pi 与所有 CSFV 的距离都大于 ε，产生新的 CSFV，加入
      CSFVset */
      { CSFV.C = Pi.C;CSFV.CS = Pi.C; CSFV.N = 1;
      CSFVset = CSFVset ∪ CSFV; }
  }
  for each CSFV generated above{
   // 计算候选特征空间数据点的重心坐标值
   CSFV'.C = CSFV.CS/CSFV.N;
   // 不在误差范围内，调整 CSFV 的属性值
   if (dist(CSFV.C,CSFV'.C) > a × ε)
     { CSFV.C = CSFV'.C;CSFV.CS = 0;
     CSFV.N = 0;}
   else {
    // 根据密度阈值进行筛选，得到 SFVset
    if (CSFV.N > = MinPts)
      { SFV.C = CSFV.C;
      SFV.N = CSFV.N;
      SFV.No = 0;
      SFVset = SFVset ∪ SFV;
      }
    delete CSFV from CSFVset;
    }
   }
  }
```

GSFV 算法中误差因子 a 是控制 while 循环的终结因素。误差因子越小，产生的特征向量集反映数据空间几何特征越精确。我们考虑算法的循环部分，分析时间复杂度：假设该站点的数据量为 n，在生成 $CSFV$ 的过程中，$CSFV$ 的数目为 l，while 语句执行了 m 次，那么时间复杂度为 $O(mnl)$，其中 m 和 l 远小于 n，所以它的时间复杂度是近似线性的，比 DBSCAN 执行效率高。

在 DBSCAN 算法中，一个 ε 邻域内可能有多个核心对象，而本文中每个特征空间只用一个 SFV 来表示，所以 SFV 的数量远小于 DBSCAN 中的核心对象。SFV 的坐标近似于特征空间的重心坐标，密度反映该特征空间内数据点的密集程度，因此 $SFVset$ 能够反映站点密集数据空间几何特征。把这个特征向量集合发送到中心站点，作为原始数据的一个有效近似。

相对于 DBDC 算法，GSFV 避免了从核心对象中产生局

部代表对象的操作,也不需要调整 ε 数值,在近似线性时间内得到特征向量的集合,提高了效率。在 DBDC 算法中,各站点向中心站点传送局部代表对象的坐标值和 ε 数值,忽略了这个代表对象邻域的密集程度。而传送特征向量集可以使中心站点的全局聚类充分考虑到各站点的数据分布,提高了聚类的精度。

1.4　全局聚类

中心站点接收所有站点(假设共 S 个站点)的 *SFVset* 后,首先进行集合的合并操作,得到所有站点的站点特征向量 ***Globalset*** $= \cup SFV_{ij}, i = 1,2,\cdots,S$。$SFV_{ij}$ 表示第 i 个站点的第 j 个特征向量。然后采用 GGFV(Generate_Global_Feature_Vector)算法获得全局特征向量 ***GFV***。最后根据定理 1 对距离小于 2ε 的全局特征点进行连接操作,得到聚类结果并将经过聚类标注的 *SFV* 发送到各站点。

1.4.1　产生全局特征向量

算法 GGFV

输入 Globalset:所有站点的特征向量集; ε:空间半径; *MinPts*:密度阈值;a:误差因子;S:站点个数

输出 *GFVset* :全局特征向量集

```
GFVset = ∅;
// 初始化, 随机选择一个站点特征向量
CGFVset = {(SFV.C.0,0)}
while (CGFVset ≠ ∅){
  for (int i = 1;i <= Globalset.size;i++) {
    for each CGFV
      with dist(SFV_i.C,CGFV.C) <= ε{
      CGFV.CS = CGFV.CS + SFV_i.C × SFV_i.N;
      CGFV.N = CGFV.N + SFV_i.N;}
    /*SFV_i 与所有 CGFV 的距离大于 ε,产生新的 CGFV,加入
    CGFVset*/
    if no CGFV
      that dist(SFV_i.C ,CGFV.C) <= ε
      {CGFV.C = SFV_i.C;
      CGFV.CS = SFV_i.C × SFVi.N;
      CGFV.N = SFV_i.N;
      CGFVset = CGFVset ∪ CGFV;}
  }
  for each CGFV generated above{
   // 调整 CGFV 的值,以逼近重心位置
   CGFV'.C = CGFV.CS/CGFV.N;
   if ( dist(CGFV.C,CGFV'.C) > a × ε)
     { CGFV.C = CGFV'.C;
     CGFV.CS = 0;
     CGFV.N = 0; }
   else {if (CSFV.N >= S × MinPts)
           { GFV.C = CGFV.C;
           GFV.N = CGFV.N; GFV.No = 0;
           GFVset = GFVset ∪ GFV;
     }
     delete CGFV from CGFVset;
     }
   }
}
```

GGFV 算法和 GSFV 算法的区别在于,GGFV 不仅考虑站点特征向量 ***SFV*** 本身的坐标信息,还考虑站点特征空间的密集程度,即密度 N 作为 ***SFV*** 参与计算的权重。因此 CGFV 代表位于相应特征空间内所有站点的数据,这比 DBDC 算法单纯地考虑代表对象的坐标来计算密度要精确很多。因为 DBDC 算法在全局聚类时把局部代表对象当作一个点来处理,而这里通过 ***SFV*** 掌握了特征空间所有点的分布信息。*CGFV.N* 值反映了所有站点数据属于 CGFV 相应特征空间的数据个数,所以可以取最小数目 $M = S \times MinPts$,其中 S 是站点个数,过滤密度小于 M 的 CGFV,得到全局特征向量 ***GFV***。

1.4.2　聚类

根据定理 1,比较每两个 *GFV.C* 的距离,当小于 2ε 时,对 *GFV.No* 赋予相同的类标识。从而得到聚类结果,设有 m 个聚类 $GC_1, GC_2, \cdots, GC_m$。若一全局特征向量 ***GFV*** 属于全局聚类 GC_k,则 *GFV.No* $= k$,其中 k 为该聚类的唯一标识,即类标号。对所有站点 ***SFV*** 进行类标识,若它位于一个 ***GFV*** 的特征空间,则 *SFV.No* = *GFV.No*。若不属于任何 ***GFV*** 的特征空间,则保持 *SFV.No* $= 0$,说明它在全局聚类中是噪声。最后将所有 *SFV.No* $\neq 0$ 的 ***SFV*** 发送到各个站点,以便站点把数据进行分类。

1.5　站点数据的分类

站点接收到经过聚类标识的特征向量 ***SFV***,扫描数据集,对每个数据点 P 进行聚类标识。若数据点属于某 ***SFV*** 的特征空间,则 P 属于类标号为 *SFV.No* 的全局聚类。若数据点 P 不在任何 ***SFV*** 的特征空间,则标识为噪声。

2　性能评估

为了测试算法的性能,我们在 10Mb 的局域网中,用 7 台 PC 机构成分布式站点,各台 PC 机配置均为:Intel 1.8G/512MB,Windows2000(Server 版),所用代码均用 Visual C++(6.0)实现。实验中所使用的仿真数据产生器是根据 DataGen[8] 改进而成。用户可以通过输入参数来控制产生数据集的结构与大小。参数包括数据集的大小、维数和各维上的取值范围(实验中为[0,100])等。将 DCBFV 算法与 DBDC 算法进行聚类精度和执行效率的比较,结果如图 1,图 2 所示。其中图 1 中的纵坐标精度是相对于集合所有站点数据进行聚类的效果而言的。DCBFV 全局聚类时充分考虑了各站点的数据分布,所以其精度整体比 DBDC 高。GSFV 算法和 GGFV 算法比 DBSCAN 算法执行效率高,特征向量的网络传输开销较少,所以 DCBFV 的执行效率较高,随着数据量的增大,这种优越性越明显。

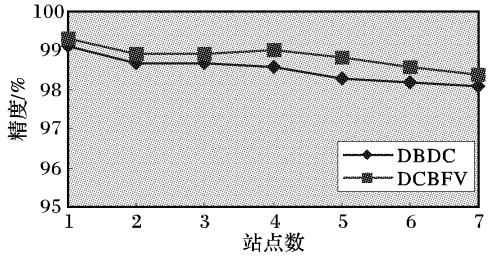

图 1 精度比较

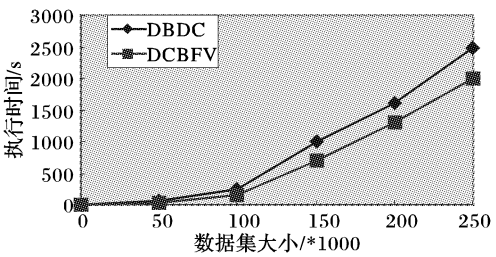

图 2　执行效率比较

3 结语

本文在分布式环境下,提出了一种基于特征向量的高效分布式聚类算法 DCBFV。该算法根据数据分布的空间特征,引入特征点、特征空间、特征向量的概念,在此基础上进行局部和全局两个层次的聚类分析。首先在各个站点采用 GSFV 算法产生站点特征向量,将站点特征向量发送到中心站点进行全局聚类,在全局聚类时充分考虑了各站点数据的分布特性,得到了比较精确的结果,最后进行数据点的类标识。DCBFV 适合多站点、大规模数据挖掘。我们的理论分析和实验结果也证明了这种算法的优越性。

DCBFV 算法适合分布式环境,它能有效解决多站点低维数据的聚类。下一步的工作是测试它在高维情况下的性能。此外,当站点和数据量都十分庞大的情况下,我们考虑对站点特征向量进行筛选,然后传送给中心站点,以此作相应算法的设计和研究。

参考文献:

[1] AGRAWAL R, GEHRKE J, GUNOPULOS D, *et al*. Automatic subspace clustering of high dimensional data for data mining application[A]. Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data[C]. New York: ACM Press, 1998.94 - 105.

[2] HINNEBURG A, KEIN DA. An efficient approach to clustering in large multimedia databases with noise[A]. Proceedings. of the 4th International Conference on Knowledge Discovery and Data Mining (KDD'98) [C]. New York: AAAI Press, 1998.58 - 65.

[3] ESTER M, KRIEGEL H-P, SANDER J, *et al*. A density-based algorithm for discovering clusters in large spatial databases with noise [A]. Proceedings of the 2nd International Conference On Knowledge Discovery and Data Mining (KDD'96) [C]. Portland: AAAI Press, 1996. 226 - 231.

[4] 马帅, 王腾蛟, 唐世渭, 等. 一种基于参考点和密度的快速聚类算法[J]. 软件学报, 2003, 14(6): 1089 - 1095.

[5] JANUZAJ E, KRIEGEL H-P, PFEIFLE M. Density Based Distributed Clustering[A]. Proceedings. of the 9th International Conference on Extending Database Technology[C]. Springer, 2004. 88 - 105.

[6] JANUZAJ E, KRIEGEL H - P, PFEIFLE M. Scalable Density - Based Distributed Clustering[A]. Proceedings of 8th European Conference on Principles and Practice of Knowledge Discovery in Databases(PKDD) [C]. Springer, 2004. 231 - 244.

[7] BECKMANN N, KRIEGEL H-P, SCHNEIDER R, *et al*. The R * - tree: An Efficient and Robust Access Method for Vector and Rectangles[A]. Proceedings ACM SIGMOD International Conference on Management of Data (SIGMOD'90) [C]. Atlantic City, NJ, New York: ACM Press, 1990. 322 - 331.

[8] Dataset Generator(DatGen) [CP/OL]. http://www.data.setgenerator.com/, 2005.

(上接第 372 页)

清洁生产评审共有 10 个指标:原材料利用率,权重 12;生产工艺,权重 12;产品销售、使用方法,权重 10;污染物产生、回收及削减率,权重 18;产品报废,权重 8;职工健康,权重 8;生产管理,权重 9;过程控制,权重 9;区域环境影响,权重 10;公众意见,权重 4。评价方法:$E = \sum A_i W_i$,其中,E 是评估总分,A_i 是指标 i 的评分值,W_i 是指标 i 的权重。每个指标的等级分值范围为 0 ~ 1.00,提示等级分段关系为:1)本行业国内中下水平,[0,0.25];2)本行业国内平均水平,[0.25,0.5];3)本行业国内先进水平,[0.5,0.75];4)本行业国际先进水平,[0.75,1.00]。在分段内的具体分值由评估者根据上述提示确定。

A表初始状态

指标名称	分项计分方法	分项分值	类型
原材料利用率	12 * x[1,2]		0
…			…
公众意见	4 * x[10,2]		0
总分	A(1,3) + … + A(10,3)		2

X表初始状态

指标名	状态
原材料利用率	
…	
公众意见	

注:1、x数组是推理机中镜像数组,专用于镜像X表,x[i,2]对应x表第2列。
2、A 数组是推理机中镜像数组,专用于镜像A表,A[i,3]对应A表第3列。

图4 系统初始状态

所以,A 表中有 11 行,其中前 10 行对应上述 10 个指标,第 11 行为总分。由于指标初始状态由主观判断,故无须第二类表,A 表中的备注字段前 10 行标注为 0。X 表中共有 10 行,其中状态列数据待装入,在本例中实际上是评估者给出的相应指标的评分。X 表数据装入完成后,推理机起动,读出 X 表数据,查询 A 表的计分方法,计算分值,得出总分,评估过程结束。各表初始状态见图 4。

6 结语

本文方法不仅适用于环境评估系统的知识库设计,也适用于一般评估系统的知识库设计,比如企业信用等级评估系统。本方法所依赖的镜像数组技术将关系数据库的标识粒度细化到数据单元,使镜像技术下的数据表初步具备了电子表格的知识表示能力。但是,该技术要求先于公式确定数据表长度。

参考文献:

[1] 许建潮, 胡明. 基于关系模式的知识表示方法[J]. 小型微型计算机系统, 1995, 16(6): 53 - 55.

[2] 吴海桥, 刘毅, 丁运亮, 等. 基于关系数据库的知识库的建立[J]. 微型电脑应用, 2001, 17(11): 52 - 54.

[3] 李晓强, 崔德光. 基于关系数据库的知识库结构设计[J]. 计算机工程与应用, 2001, (24): 102 - 103, 147.

[4] 许云, 樊孝忠. 在专家系统中利用关系数据库来表达知识[J]. 计算机工程与应用, 2003, (22): 91 - 93, 138.

[5] 谢晓方, 姜震. 基于关系数据库的知识库系统设计方法[J]. 微计算机应用, 2004, 25(2): 227 - 230.

[6] 雷英杰, 邢清华, 王涛, 等. 人工智能(AI)程序设计(面向对象语言)[M]. 北京: 清华大学出版社, 2005. 45 - 85.