基于分布式的大数据集聚类分析

贾俊芳,张日权
JIA Jun-fang,ZHANG Ri-quan

大同大学 数学与计算机学院,山西 大同 037009
Mathematics and Computer Institute,Datong University,Datong,Shanxi 037009,China
E-mail:jiajunfang816@163.com

JIA Jun-fang,ZHANG Ri-quan.Large data sets clustering analysis based on distribution.Computer Engineering and Applications,2008,44(28):133-135.

Abstract: In order to improve the efficiency we propose a distributed clustering algorithm based on large data sets.Namely data is randomly divided into several subsets without clustering all the data at a time,then we cluster all the subsets at the same time.At last we combine the genus.Experiment results show that most of time the result is the same as using traditional clustering algorithm,and it improves the clustering speed greatly.

Key words: clustering analysis;distribute;large data sets

摘 要: 为了提高聚类效率提出了一种基于分布式的大数据集聚类算法。该方法并不是一次性对所有的数据进行聚类,而是将大数据集随机分成若干个子集,对每个子集同时进行聚类,最后进行类的合并。实验结果表明大多数情况下该方法和传统的一次性聚类的结果一致,而且极大地提高了聚类的速度。

关键词: 聚类分析;分布式;大数据集
DOI: 10.3778/j.issn.1002-8331.2008.28.045 **文章编号:** 1002-8331(2008)28-0133-03 **文献标识码:** A **中图分类号:** TP18

1 概述

在过去几年,人工智能的研究取得了长足的进展,然而还有很多重要的问题没有得到满意的解决。已有的聚类算法有传统的硬 K-means 聚类算法和广泛应用在模糊聚类的 FCM 软聚类算法,这些算法对大数据集聚类时效率非常低[1],因此,提高速度显得尤为重要。FCM 算法不能很好地测量大数据集,而且也存在这种情况:数据是被分布在不同的地理位置[2]。鉴于此,本文提出了一种基于分布式的大数据集聚类算法。

聚类就是按照事物间的相似性进行区分和分类的过程,在这一过程中没有教师指导,因此是一种无监督的分类。聚类分析则是通过无监督训练将样本按相似性分类,把相似性大的样本归为一类,占据特征空间的一个局部区域,而每个局部区域的聚类中心又起着相应类型的代表的作用。聚类分析一方面可以作为一种有效的信息压缩与提取手段,另一方面又往往是其他模式识别的基础。

传统的聚类分析[3]是一种硬划分,它把每个待辨识的对象严格地划分到某个类中,具有非此即彼的性质,因此这种分类的类别界限是分明的。而实际上大多数对象并没有严格的属性,它们在性态和类属方面存在着中介性,适合进行软划分。Zadeh[4]提出的模糊集理论为这种软划分提供了有力的分析工具,人们开始用模糊的方法来处理聚类问题,并称之为模糊聚类分析。进行模糊聚类的最广泛应用的算法是 FCM 算法[5]。

2 FCM 算法

设有限集 $X=\{x_1,x_2,\cdots,x_n\}$ 属于 P 维欧几里德空间 R^P,即 $x_j\in R^P,j=1,2,\cdots,n$。FCM 算法中的目标函数采用的是基于欧氏距离,该目标函数为:

$$J=\sum_{i=1}^{c}\sum_{j=1}^{n}\mu_{ij}^{m}\|x_j-\omega_i\|^2 \tag{1}$$

其中,n 为样本数,c 为给定的类别数,并且 $1<c<n$,m 为加权幂指数,它影响隶属度矩阵的模糊度,ω_i 为第 i 类的聚类中心,μ_{ij} 表示样本 j 属于第 i 类的程度,满足 $\sum_{i=1}^{c}\mu_{ij}=1,j=1,2,\cdots,n$。

FCM 算法就是要求使 J 达到最小值的聚类结果。因此将 J 分别对 μ_{ij} 和 ω_i 求导,令它们的倒数为 0,并代入条件 $\sum_{i=1}^{c}\mu_{ij}=1$,得:

$$\omega_i=\sum_{j=1}^{n}(\mu_{ij})^m x_j\Big/\sum_{j=1}^{n}(\mu_{ij})^m,\ i=1,2,\cdots,c \tag{2}$$

$$\mu_{ij}=\left(\frac{1}{\|x_j-\omega_i\|_G^2}\right)^{\frac{1}{m-1}}\Big/\sum_{k=1}^{c}\left(\frac{1}{\|x_j-\omega_k\|_G^2}\right)^{\frac{1}{m-1}},\quad i=1,2,\cdots,c;j=1,2,\cdots,n \tag{3}$$

其中,$\|x_j-\omega_i\|_G^2=(x_j-\omega_i)^{\mathrm{T}}\boldsymbol{G}(x_j-\omega_i)$,$\boldsymbol{G}$ 为一个 $P\times P$ 维的对称

基金项目: 山西省自然科学基金(the Natural Science Foundation of Shanxi Province of China under Grant No.2007011014)。
作者简介: 贾俊芳(1976-),女,讲师,研究方向:模糊推理,数据挖掘;张日权(1967-),男,教授,博导,研究方向:数理统计和数据挖掘。
收稿日期: 2007-11-20 **修回日期:** 2008-02-28

正定矩阵。

FCM 算法是通过对式(2)和式(3)进行迭代来完成的。当 J 收敛到极小值时,就得到了最终的聚类结果,即得到了各类的聚类中心和各个样本对于各类的隶属度。

3　分布思想

将一个规模为 n 的问题分解为 k 个规模较小的子问题,这些子问题相互独立且与原问题相同。递归地解这些子问题,然后将各子问题的解合并得到原问题的解。它的一般算法设计模式如下:

```
Divide_and_conquer(p)
{if(|p|<=n0)adhoc(p);
Divide p into smaller subinstances p1,p2,…,pk;
for(i=1,i<=k;i++)
yi=Divide_and_conquer(pi);
return_merge(y1,y2,…,yk);}
```

4　基于分布式的大数据集聚类算法

4.1　算法描述及流程图

算法过程描述如下:

(1)给定数据集 $A=\{a_1,a_2,\cdots a_n\}$;

(2)将数据集 A 分成若干个子集 $A_1,A_2,\cdots,A_p$;

(3)对子集 $A_1,A_2,\cdots,A_p$ 同时进行聚类,分别得到的聚类中心数为 $m_1,m_2,\cdots,m_p$;

(4)if $|m_1+m_2+\cdots+m_p|\geqslant n_0$($n_0$ 为问题规模的阈值),then 把 $m_1+m_2+\cdots+m_p$ 个聚类中心看成集合 A,转到(2);else 转到(5);

(5)把 $m_1+m_2+\cdots+m_p$ 个数据进行一次性聚类;

(6)类的合并:第(5)步结束后,if 聚类中心 x_1 和 x_2 聚为一类,而在第(3)步结束后 c_1 和 c_2 分别是以 x_1 和 x_2 为聚类中心的类,then 就把类 c_1 和 c_2 合并为一类。

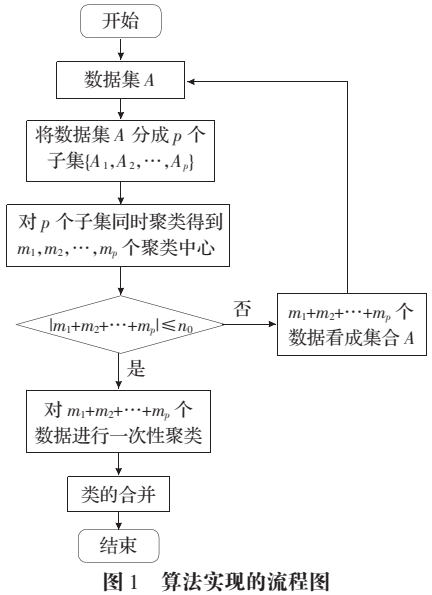

图 1　算法实现的流程图

4.2　性能分析

本文提出的算法能有效地提高大数据集的聚类速度。首先与传统的 K-means 聚类算法作比较，在 K-means 聚类算法中每一次迭代中需要执行的步骤如下:

(1)对于每一个 $k\neq i$,如果 $\|x_j-c_i\|^2\leqslant\|x_j-c_k\|^2(k=1,2,\cdots,K)$,则 $\mu_{ij}=1$,否则 $\mu_{ij}=0,j=1,2,\cdots,n$。

(2)$c_k=1/|G_k|\sum_{j,x_j\in G_k}x_j$($|G_k|$为属于第 k 类的规模数)。

如果一共有 n 个数据,且分为 k 类,那么每一次迭代在第一步中需要比较 $n\times k$ 次,若需要迭代 L 次,那么一共需要比较 $L\times n\times k$ 次,在第二步中需要计算 $k\times L$ 次,所以对 n 个数据聚类一共需要的时间 $T(n)=o(L\times k\times n+k\times L)$。因为 n 很大,所以 $L\times k\times n$ 会很大,这样执行的效率会非常低。而本文提出的方法采用分布式,如果一个分布式将规模为 n 的问题分成 s 个规模为 n/m 的子问题来解决,那么可得到执行时间 $t(n)=o(n^{\log_m^s})$。如果将 n 个数据平均分为 s 个子集,那么每个子集就有 n/s 个数据,此时 $s=m$,那么 $t(n)=o(n)$,由于 $o(n)<o(L\times k\times n+k\times L)$,可见此时的时间复杂度降低了很多,所以极大地提高了聚类的速度。

然后再与 FCM 算法作比较,FCM 算法在每一次迭代需要执行的步骤为:

$$(1)\omega_i=\sum_{j=1}^{n}(\mu_{ij})^m x_j\Big/\sum_{j=1}^{n}(\mu_{ij})^m,\ i=1,2,\cdots,c$$

$$(2)\mu_{ij}=\left(\frac{1}{\|x_j-\omega_i\|_G^2}\right)^{\frac{1}{m-1}}\Big/\sum_{k=1}^{c}\left(\frac{1}{\|x_j-\omega_k\|_G^2}\right)^{\frac{1}{m-1}}$$

$i=1,2,\cdots,c;j=1,2,\cdots,n$

如果需要聚类的数据为 n,且为 c 类,那么在第(2)步中对于每一个数据点需要计算 c 次，所以一共需要计算 $n\times c$ 次,而第 1 步需要计算 c 次。如果一共迭代 L 次,那么聚类这 n 个数据所要的时间为 $T(n)=o(L\times c\times n+L\times c)$,而 n 很大,所以执行速度很慢。而采用本文提出的方法,如果把数据 n 平均分成 k 个子集,那么每个子集的数据量为 n/k 个,同样可得到此时的执行时间为 $t(n)=o(n)$,而 $o(n)<o(L\times c\times n+L\times c)$,所以降低了时间复杂度,极大地提高了效率。

5　实验仿真

实验选取了一部分 iris 数据，且使用了 petal length 和petal width 两个属性。

表 1　选取的 iris 数据

第 1 子集	1.3 0.2;1.7 0.4;1.4 0.3;1.4 0.2;1.5 0.1;1.5 0.2;1.6 0.2;1.2 0.2; 1.4 0.1;1.1 0.1
第 2 子集	4.7 1.4;4.5 1.5;4.9 1.5;4 1.3;4.6 1.5;4.5 1.3;4.7 1.6;3.3 1; 4.6 1.3;3.9 1.4;3.5 1;4.2 1.5;4 1;3.6 1.3
第 3 子集	5.9 2.1;5.6 1.8;5.8 2.2;6.6 2.1;6.3 1.8;5.8 1.8;6.1 2.5;6.4 2; 5.3 1.9;5.5 2.1;6.1 1.9;5.3 2.3;5.5 1.8;6.7 2.2;6.9 2.3

对表 1 中所有的数据进行一次性聚类(图 2),设置聚类中心数为 3。聚类结果得到 3 类,其中每类所包含的数据正好对应于表 1 中的每个子集。

表 2　对集合 A 聚类后的结果

聚类中心	属于该类的数据
(1.408 9 0.198 8)	1.3 0.2;1.7 0.4;1.4 0.3;1.4 0.2;1.5 0.1; 1.5 0.2;1.6 0.2;1.2 0.2;1.4 0.1;1.1 0.1
(3.456 5 1.066 8)	3.3 1;4 1.3
(4.622 7 1.438 8)	4.7 1.4;4.5 1.5;4.9 1.5;4.6 1.5;4.5 1.3; 4.7 1.6;4.6 1.3

表 3 对集合 B 聚类后的结果

聚类中心	属于该类的数据
(3.831 4 1.235 6)	3.9 1.4;3.5 1;4.2 1.5;4 1;3.6 1.3
(5.607 2 1.994 7)	5.9 2.1;5.6 1.8;5.8 2.2;5.8 1.8;5.3 1.9;5.5 2.1;5.3 2.3;5.5 1.8
(6.514 6 2.113 1)	6.6 2.1;6.3 1.8;6.1 2.5;6.4 2;6.1 1.9;6.7 2.2;6.9 2.3

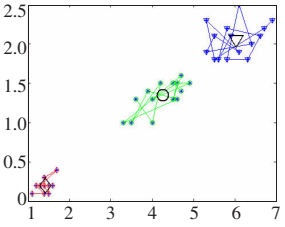

图 2 对表 1 的数据进行一次性聚类结果

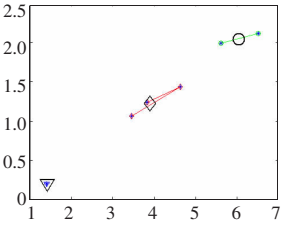

图 3 对子集 C 进行的聚类

在利用分布式聚类时,把原始数据分为 2 个或 3 个子集,然后再对每个子集聚类时设置聚类中心数为 $number$=3,以及在最后的一次聚类设置它的聚类中心也为 3。

下面是其中的一次实验:原始的数据集被分为 2 个子集 A={1.3 0.2;1.7 0.4;1.4 0.3;1.4 0.2;1.5 0.1;1.5 0.2;1.6 0.2;1.2 0.2;1.4 0.1;1.1 0.1;4.7 1.4;4.5 1.5;4.9 1.5;4 1.3;4.6 1.5;4.5 1.3;4.7 1.6;3.3 1;4.6 1.3},B={3.9 1.4;3.5 1;4.2 1.5;4 1;3.6 1.3;5.9 2.1;5.6 1.8;5.8 2.2;6.6 2.1;6.3 1.8;5.8 1.8;6.1 2.5;6.4 2;5.3 1.9;5.5 2.1;6.1 1.9;5.3 2.3;5.5 1.8;6.7 2.2;6.9 2.3}。

将两个子集聚类后的聚类中心放在一起形成一个集合 C={1.408 9 0.198 8;3.456 5 1.066 8;4.622 7 1.438 8;3.831 4 1.235 6;5.607 2 1.994 7;6.514 6 2.113 1}。该集合已包含较少的数据,因此没有必要去递归,所以直接对 C 集合进行聚类,结果如图 3。

最后进行类的合并。子集 C 聚类后的结果:(1.408 9 0.198 8)为一类;(3.456 5 1.066 8)和(3.831 4 1.235 6)以及(4.622 7 1.438 8)为一类;(5.607 2 1.994 7)和(6.514 6 2.113 1)为一类。所以集合 A 聚类后以(1.408 9 0.198 8)为聚类中心的这个类为最后的一类;把集合 A 聚类后以(3.456 5 1.066 8)和(4.622 7 1.438 8)为聚类中心的这两个类以及集合 B 聚类后以(3.831 4 1.235 6)为聚类中心的类归为一类;集合 B 聚类后以(5.607 2 1.994 7)和(6.514 6 2.113 1)为聚类中心的这两个类归为一类。可以得到与一次性聚类一致的结果。

利用该方法连续做实验 20 次,其中只有一次产生分歧。产生分歧的结果是将第 2 类中的(4.9 1.5)分到了第 3 类。发现(4.9 1.5)正好是在第 2 类的边界。可见在边界处的点容易被分出去。

6 结论

本文提出了一种基于分布式的大数据集聚类方法,首先通过理论分析可知利用该方法能提高对大数据集聚类的速度,因为把大数据集分成了几个子集后,每个子集包括了较少的数据,而且这些子集可以同时进行聚类。

然后通过实验可得:利用该方法在大多数情况下与一次性聚类的结果一致,产生分歧的情况是由于类的边界。

参考文献:

[1] Davidson I,Satyanarayana A.Speeding up k-means clustering by bootstrap averaging[C]//the Workshop on Clustering Large Data Sets,IEEE ICDM 2004.

[2] Strehl A,Ghosh J.Clusters ensembles-a knowledge reuse framework for combining multiple partition.Journal of Mathine Learning Research,2002;3,583.

[3] Zhang Yuanquan,Rueda L.A geometric framework to visualize fuzzy-clustered data[C]//IEEE Proceedings of the XXV International Conference of the Chilean Computer Science Society,2005.

[4] Hoppner F.Speeding up fuzzy c-Means:using a hierarchical data organization to control the precision of membership calculation[J].Fuzzy sets and Systems,2002,128:30-50.

[5] 郑岩,黄蓉怀,站晓苏.基于遗传算法的动态模糊聚类[J].北京邮电大学学报,2005,28(1).

(上接 124 页)

响应速度。实验表明,SMVS 算法能够有效提高空间数据仓库的查询效率,但用于调整物化视图集的预留空闲空间对算法的性能影响很大,其大小很难确定,这将是下一步要研究的工作之一。

致谢 感谢东南大学计算机科学与工程学院的孙志挥教授对论文的悉心指导,从而使论文能得以顺利完成。

参考文献:

[1] Harinarayan V,Rajaraman A,Ullman J D.Implementing data cubes efficiently[C]//Proceeding of ACM SIGMOD Int'l Conf on Management of Data,New York,1996:205-227.

[2] Gupta H.Selection of views to materialize in a data warehouse[C]//Afrati F N,Kolaitis P G.Proceeding of the 6th ICDT,Heidelberg,1997:98-112.

[3] Horng J T,Chang Y J,Liu B J.Applying evolutionary algorithms to materialized view selection in a data warehouse[J].Soft Computing,2003,7(8):574-581.

[4] Kotidis Y,Roussopoulos N.DynaMat:a dynamic view management system for data warehouses[C]//Delis A,Faloutsos C,Ghandeharizadeh S.Proceeding of the 1999 ACM SIGMOD Int'l Conf on Management of Data,New York,1999:371-382.

[5] Shekhar S,Chawla S.Spatial databases[M].[S.l.]:Prentice Hall,2003.

[6] Stefanovic N,Han J,Koperski K.Object-based selective materialization for efficient implementation of spatial data cube[J].IEEE Transactions on Knowledge and Data Engineering,2000,12(6):938-958.

[7] Yu S M,Vijayalakshmi A,Nabil A.Selective view materialization in a spatial data warehouse[C]//Lecture Notes in Computer Science 3589:Proceeding of 7th International Conference on Data Warehousing and Knowledge Discovery,Copenhagen,2005:157-167.

[8] Li J J,Wang Y,Liu R Q.Selection of materialized view based on information weight and using Huffman-tree on spatial data warehouse[C]//Proceeding of the First International Conference on Innovative Computing,Information and Control,Beijing,2006:71-74.

[9] Theodoridis Y,Stefanakis E,Sellis T.Efficient cost model for spatial queries using r-trees[J].IEEE Transactions on Knowledge and Data Engineering,2000,12(1):19-32.