计算机科学 2006Vol.33№.1

具有输入知识的高维数据聚类算法研究*)

吴红艳[1] 王蔚韬[1] 文俊浩[2] 何光辉[1]

(重庆大学计算机学院 重庆 400030)[1] (重庆大学软件学院 重庆 400030)[2]

摘 要 针对目前聚类算法没有充分地利用输入知识,不便于知识的学习和增长的情形,提出在高维数据集的情况下,恰当地利用输入知识可以更准确有效地发现聚类,提出聚类的相关维集的概念,分析输入知识的特点,对带有输入知识的高维聚类算法进行研究,指导聚类的学习过程。

关键词 聚类,聚类算法,高维数据集,输入知识,相关维集

Research on Clustering Algorithm of Hi-dimensional Dataset with Input Knowledge

WU Hong-Yan[1] WANG Wei-Tao[1] WEN Jun-Hao[1,2] HE Guang-Hui[1]

(College of Computer Science, Chongqing University, Chongqing 400030)[1]

(College of Software Engineering, Chongqing University, Chongqing 400030)[2]

Abstract Due to ignorance of available input knowledge current clustering algorithms are not beneficial to the learning and increment of knowledge. With input knowledge clusters can be discovered efficiently and properly. Based on the characteristics of input knowledge the relevant dimension sets of cluster are introduced in this paper. And then clustering algorithm of hi-dimensional dataset with input knowledge is researched to supervise the clustering process.

Keywords Clustering, Clustering algorithm, Hi-dimensional dataset, Input knowledge, Relevant dimension set

1 引言

聚类所具有的无指导学习能力使它具有广泛的应用空间,例如模式识别、图像处理等。然而,其无指导学习性也极大地限制聚类算法的效率和应用前景,不利于知识的学习和增长。例如,人类所具有的专家知识或者普遍性知识不能充分地应用到聚类算法中,同时,经过聚类或者其他各种方法进行学习后所获得的新知识不能作为指导性知识来指导聚类算法进行学习,不利于知识的应用和进行增长式学习。如何把学习所获得的知识或者专家知识应用到聚类中,指导聚类进行进一步学习,对于知识的增长具有非常重要的意义。

对高维数据的处理是聚类的一个重要应用领域。但是,在实际的高维数据应用中,如果需要对某类具有上百个属性的对象进行聚类,使用相似性函数考察对象的所有属性时,两个对象之间的相似性可能会非常低。从而难以得到理想的聚类结果。到目前为止,有很多文献对如何进行高维对象之间的聚类进行了研究,例如采用主成分分析法等[1],不过,如果要研究的聚类的相关属性占总属性的不到10%或者更低,采用纯粹的无监督的聚类方法就很难找出正确的聚类结果。本文对如何提取高维数据中的相关维进行分析,并对在具有少量的输入知识的情况下,如何指导聚类学习进行研究。

2 聚类的相关理论

2.1 聚类的定义

聚类是一个把多维输入数据空间划分为多组相似对象的过程。文[2]对聚类定义如下:数据点 $x_i=(x_{i1},\cdots,x_{id},)\in A$, x_i 的每个属性(特征或维度) x_{il} 既可以是数值型的,也可以是枚举型的。数据集 X 相当于一个 Nxd 矩阵。假设数据集 X 中有 N 个对象 x_i, $i=1,\cdots,N$。聚类的最终目的是把数据集 X 划分为 k 个分割 C_i, $i=1,\cdots,k$,也可能有些对象不属于任何一个分割,这就是噪声 $C_{outliers}$。所有这些分割与噪声的并集就是数据集 X,并且这些分割之间没有交集,即 $X=C_1\cup\cdots U_k\cup C_{outliers}$, $C_i\cap C_j=\Phi$,这些分割 C,就是聚类。

2.2 带输入知识的聚类

带输入知识的聚类是指使用已经具备的领域知识指导聚类学习过程,也可以称为半指导式聚类。它不同于分类。分类所使用的类别知识可以明确地对对象进行分类,而带输入知识的聚类所使用的知识却不足以将对象进行分类,例如,指导性知识可能只涉及到某一类,不能覆盖分类所需要的所有类知识;另外,指导性知识可能和分类不相关,这些知识不是和对象所属的类别相关的知识。根据知识输入的时间以及如何使用知识来影响聚类过程,带输入知识的聚类方法也会有所不同。

标识了的对象是最简单的输入知识。在某些情况下,用户可能并不知道对象的确切的类标识,但是知道对象应该或者不应该属于某个聚类。还有某些文献建议使用分类规则[3]、相似对象示例作为输入知识[4],甚至某个对象不属于某个聚类也可以作为输入知识[5]。

知识可以在聚类过程的不同时期输入。可以在开始聚类之前输入知识指导聚类过程;也可以在聚类结束之后,使用输入的知识对所得聚类进行评价,以便指导下一次的聚类;也有一些算法在聚类的过程中和用户进行交互,以便在最合适的

*)基金项目:重庆市自然科学基金支持项目(CSTC,2004BB2182)。吴红艳 硕士研究生,主要研究方向:数据挖掘及信息支持;王蔚韬 副教授,主要研究方向:数据挖掘及信息支持;文俊浩 副教授,博士,主要研究方向:软件工程、数据挖掘;何光辉 主要研究方向:数据挖掘及模式识别。

时候让用户输入知识指导聚类过程。

使用输入知识的方法也可以有所不同，例如，指导聚类的种子的选取，建议某些对象归入同一个聚类，或者使用输入知识来修改聚类的目标函数、相似性函数以及距离矩阵等。

2.3 划分算法

由于本文所使用的算法和划分算法相关，所以下面对主要的划分聚类算法略作介绍。

划分聚类算法把数据点集分为 k 个划分，每个划分作为一个聚类。它一般从一个初始划分开始，然后通过重复的控制策略，使某个准则函数最优化，而每个聚类由其质心来代表(K-means 算法)，或者由该聚类中最靠近中心的一个对象来代表(K-medoids 算法)[6]。

K-medoids 算法首先随机选择 k 个对象，每个对象代表一个聚类的中心。对于其余的每一个对象，根据该对象与各聚类中心之间的距离，把它分配到与之最相似的聚类中。然后，计算每个聚类的中心，并用最靠近中心的对象来代表该聚类，重复上述过程，直到准则函数会聚。通常采用的准则函数是平方误差准则函数(squared-error criterion)，即 $E=\sum_{i=1}^{k}\sum_{p\in C_i}|p-k_i|^2$，$E$ 是一个数据集中所有对象的误差平方和，P 是一个对象，k_i 是聚类 C_i 的中心。该算法的具体步骤如下：

① 从数据集中选择 k 个中心 $k_1,k_2,\cdots,k_k$ 作为初始的聚类中心；

② 把每个对象分配到与之最相似的聚类。每个聚类用其中所有对象的均值来代表，“最相似”就是指距离最小。对于每个点 V_i，找出一个中心 k_j，使它与其间的距离 $d(V_i,k_j)$ 最小，并把 V_i 分配到第 j 组；

③ 把所有的点都分配到相应的组之后重新计算每个组的中心 k_j；

④ 循环执行第②步和第③步，直到数据的划分不再发生变化。

该算法具有很好的可伸缩性，其计算复杂度为 $O(nkt)$，其中，t 是循环的次数。

3 带有输入知识的高维聚类算法

在实际的高维数据应用中，如果考察数据对象的所有维，两个对象之间的相似性可能会非常低，而在现实应用中，并不是所有维都相似的对象才能属于同一个聚类，故而同时考察数据集的所有维难以得到理想的聚类结果。这里针对某些特定的维进行研究，只要对象在这些维上的投影相近，则认为对象属于同一个聚类。这些和某个聚类相关的维的集合就称为该聚类的相关维集。

3.1 聚类的相关维集

数据集 D 包含 n 个对象和 d 维，该数据集可以被划分为 k 个聚类 $\{C_i\}_{i=1}^{k}$ 以及一个孤立点集。假设每个聚类是某个对应的潜在对象类的一个随机样本，并且每个聚类和一组相关维度关联，这些维形成了对象的相关子空间。用 D_j 和 C_{ij} 分别表示数据集 D 和 C_i 在某一个维 v_j 上的投影。假设 v_j 和聚类的某个子集 R_j 相关，对于每个属于 R_j 的聚类 C_i，C_{ij} 是一个局部高斯群体的随机抽样，其方差为 σ_{ij}^2。在 v_j 上投影的所有其他聚类集 $D_j-\cup_{C_i\in R_j}C_{ij}$ 是一个全局高斯群体的随机抽样，其方差为 σ_j^2，该方差值比局部方差值 σ_{ij}^2 大很多。

很显然，在聚类的一个相关子空间，属于同一个聚类的成员彼此比较靠近，而不在同一个聚类的成员则相距要远一点。也就是说，假设一个聚类有一个相关的子空间，如果一个聚类在某个维上的投影的方差比较小，那么该聚类就和这一维相关。在介绍选择相关纬集的算法之前，先引入几个公式：

$$\phi=\frac{1}{nd}\sum_{i=1}^{k}\phi_i \tag{1}$$

$$\phi_i=\sum_{V_j\in V_i}^{k}\phi_{ij} \tag{2}$$

$$\phi_{ij}=\|C_i\|-1-\frac{1}{\hat{s}_{ij}^2}\sum_{x\in C_i}(x_j-\bar{\mu}_{ij})^2$$

$$=(\|C_i\|-1)(1-\frac{s_{ij}^2+(\mu_{ij}-\bar{\mu}_{ij})^2}{\hat{s}_{ij}^2}) \tag{3}$$

其中，V_i 表示聚类 C_i 的选择维集；μ_{ij}、$\bar{\mu}_{ij}$ 和 s_{ij}^2 分别表示聚类 C_i 在 V_j 上的投影上的中值、均值以及方差；$\hat{s}_{ij}^2$ 是某个方差阈值，后面会详加解释；$\|C_i\|$ 表示 C_i 里的对象的数目。

为了提高目标函数的值，由式(3)可知，只要选择满足 $s_{ij}^2+(\mu_{ij}-\bar{\mu}_{ij})<\hat{s}_{ij}^2$ 条件的维作为相关维，形成合适的聚类，就可以提高目标函数值。$\hat{s}_{ij}^2$ 的确定方法如下：$\hat{s}_{ij}^2$ 应该大于所有的相关维的样本方差 s_{ij}^2。因为一个群体的随机样本的期待方差就是该群体的方差，如果 s_{ij}^2 不小于全局的样本方差 σ_j^2，那么聚类 C_i 里的对象就不具有相似性，所以 s_{ij}^2 肯定小于全局的样本方差 σ_j^2。因此，σ_j^2 是 $\hat{s}_{ij}^2$ 的一个最大值。这里简单地取 $\hat{s}_{ij}^2=\sigma_j^2$，由此可得选择相关维集的算法如下：

① 输入目标聚类 C_i；

② 如果 $s_{ij}^2+(\mu_{ij}-\bar{\mu}_{ij})<\hat{s}_{ij}^2$ 选择 V_j。

3.2 使用输入知识

推荐的算法在初始化时使用输入的知识。输入的知识包含两类：一类是标签的对象，使用二元组(obj. id，class label)来表示，每一组表示该对象属于某一类，用符号表示为 I_o；另一类是标签的维，使用二元组(dim. id，class label)来表示，每一组表示该维是某一类的相关维，用符号表示为 I_i。注意，每个维可能和多个类相关，但是，知识里所涉及的类可以不完全包含数据集的所有潜在类。

初始化时，算法确定聚类的种子，并把它们放入不同的种子集里。每一组种子都包含一组来自于某个真实聚类的种子，以及由这些种子评估得到的维集，即相关维集。算法创建两种不同类型的种子集：私有的和公有的。根据所输入的种子和相关维集，具有输入知识的聚类很容易形成精确的种子集，这种类型的种子集被创建为私有的种子集，该种子集只被拥有这些种子的聚类所使用。其他的多数种子所形成的种子集是公共的种子集。当聚类需要从种子集里提取自己的聚类中心时，如果该聚类有私有种子集，就随机地从它的私有种子集里提取，否则，从公共种子集里选出一个其他聚类还没有使用的种子。

另外，种子集的创建顺序也非常重要。先根据输入知识创建那些可以精确创建出来的聚类的种子集，再创建剩余的种子集，这样会比较容易些。在相关子空间，靠近那些精确创建的种子集的对象属于该种子所在聚类的可能性会比较大，创建接下来的种子集时，就不必考虑这些对象了。这样一来，创建新的种子集会比较容易，并且准确性也会相对较高。对于具有较多的输入知识的聚类，创建种子集会比较容易，准确性也较高，故而具有较多的输入知识的聚类的种子集应该优先创建。根据创建种子集的简易性以及准确性，建议种子集按照如下的顺序创建：(1)同时具有两类输入知识的聚类；(2)具有 I_o 类输入知识，即输入知识为标识的对象；(3)具有 I_i 类输入知识，即输入的知识为标识的纬度；(3)没有输入知识。

3.3 聚类中心替换策略

为了改善聚类的目标函数值，算法需要识别出来什么样的聚类是不好的聚类，并改变该聚类的中心。当算法所找到的两个聚类的中心实际上应该属于一个聚类时，就会出现不好的聚类。这两个聚类竞相提高目标函数的值，但是，却因为算法可以识别的聚类的个数在初始化时已经确定，所以有一个真正的聚类并没有被识别出来，而这个聚类所包含的对象被列入孤立点集。这种情形是不合理的。

为了解决这个问题，在算法中需要注意，是不是某个聚类在所有相关维上的函数值即 ϕ_i 特别低，或者说是出现两个特别相似的聚类。这时，就说明相关的聚类是不好的，应使用一个新的聚类中心。

3.4 带有输入知识的高维聚类算法

算法在初始化时，先确定一些聚类的种子，即潜在的聚类的中心。然后，将数据集中的每个对象分配到每一个聚类，使得目标函数的值最大，暂时使用聚类中心在 V_j 上的投影替代 $\tilde{\mu}_{ij}$ 计算目标函数的值。如果某个对象不能改善任何聚类的 ϕ_i 值，则它被认为是一个孤立点，放入孤立点集。为所有的对象分配了所在聚类之后，重新确定每个聚类的相关维集，并使用实际的中心再次计算目标函数值。如果新得到的目标函数值是目前为止最大的值，记录所形成的聚类。否则，保持前面记录的最好的聚类不变。对于新得到的聚类，重新计算每个聚类的中心。在下一次迭代中，使用新得到的聚类中心代替 $\tilde{\mu}_{ij}$，并重新为数据集里的每个对象分配聚类。重复上述操作，直到在几次迭代过程后，所得到的目标函数的值没有大的改善为止。该算法类似于 K-medoids 算法，具体的步骤如下：

① 加载数据集。

② 输入聚类个数以及相关知识(假设所输入的知识全部正确)。

③ 为每一个聚类分配一个聚类中心，并把每个对象分配到与之最相似的聚类，使得在这种情形下的目标函数值最大。

④ 选择相关维。

⑤ 如果所得聚类使得目标函数值是到目前为止的最大值，则记录该组聚类结果。否则，替换聚类中心。

⑥ 重复③，④，⑤直到所得到的目标函数在几次迭代中没有大的改善为止。

3.5 复杂性分析

算法的时间复杂度和空间复杂度分别为 $O(knd)$ 和 $O(nd)$。相对于其他同类问题的算法而言，线形的时间复杂度使得该方法对于解决大规模数据集的聚类问题更实际。

结论 本文从聚类高维数据中的应用出发，提出了带有输入知识的高维数据聚类算法，并对算法所涉及的主要技术进行讨论。本文从高维数据的聚类的相关维着手，提出正确地使用输入知识来指导聚类过程，以便把学习所获得的知识或者专家知识应用到聚类中，指导聚类进行进一步学习，利于知识的增长，扩大了聚类的应用前景。

参 考 文 献

1 Aggarwal C C, Yu P S. Finding generalized projected clusters in high dimensional spaces. In: ACM SIGMOD Intl. Conf. on Management of data, 2000.

2 Berkhin P. survey of clustering data mining techniques. http://www. accrue. com/products/ researchpapers. html

3 Talavera L, Bejar J. Integrating declarative knowledge in hierarchical clustering tasks. In:Intl. Symposium on Intelligent Data Analysis, 1999

4 Xing E P, Ng A Y, Jordan M I, Russell S. Distance metric learning, with application to clustering with side-information. In Advances in Neural Information Processing Systems 15, 2003

5 Cohn D, Caruana R, McCallum A. Semi-supervised clustering with user feedback, 2000

6 Han Jiawei, Kamber M. 数据挖掘概念与技术. 北京:机械工业出版, 2001

(上接第 133 页)

1 所示。事务时间区间在区间[1,10]和[90,100]上均匀分布时，λ 值均超过了 100，存储空间利用率也较高；在区间[1,100]上，λ 值下降明显；在区间[1,10]和[90,100]各占50%时，λ 值下降为 10.94，存储有效利用率同时下降为 69.21%。由此可见，VH 索引比较适用于状态变化的时间间隔相对接近的实体集。具体应用时，可以根据先验知识，将实体分成不同的簇，在各簇上分别建立 VH 索引，例如在仿真系统中记录实体状态的数据库可以采用这一方法。

表 1 VH 索引的性能随事务时间区间分布状况的变化

时间区间长度	[1,10]	[90,100]	[1,100]	[1,10]和[90,100]各 50%
存储利用率(%)	78.03	87.00	78.13	69.21
λ	106.95	141.92	60.81	10.94

结论与展望 上世纪 90 年代以来，研究人员对时态数据的索引技术进行了大量的研究，对时间属性的索引主要集中在树索引方面。我们在分析数据库时间属性特点的基础上，设计了一种新的基于时间属性的 Hash 索引——VH 索引。通过对其查询时间复杂度和数据库表的空间复杂度的理论分析以及实验验证，对其有效性和适用性得出了初步结论，但基于 VH 索引的时间区间相关查询及其优化方法仍需进一步研究。另外，VH 索引的聚簇特性使其适用于数据查询的并行处理，因此对于分布式应用环境下的 VH 索引的研究也很有意义。

参 考 文 献

1 Kollios G, Tsotras J. Hashing Methods for Temporal Data. IEEE Transactions on Knowledge and Data Engineering, 2002, 14(4): 902～919

2 Jensen C S, Snodgrass T. Temporal Data Management. IEEE Transactions on Knowledge and Data Engineering, 1999, 11(1): 36～44

3 Salzberg B, Tsotras V J. A Comparison of Access Methods for Time-Evolving Data, ACM Computing Surveys, 1999, 31(2): 158～221

4 Litwin W, Neimat Ma, Donovan. LH *-A Scalable, Distributed Data Structure. ACM Transactions on Database Systems, 1996, 21(4): 480～525

5 Nϕrvag K. A study of object declustering strategies in parallel temporal object database systems, Information Science, 2002 (146): 1～27

6 Nascimento M A, Dunham M H. Indexing Valid Time Database via B^+-Tree. IEEE Transactions on Knowledge and Data Engineering, 1999, 11(6): 929～947

7 Gunadhi H, Segev A. Efficient Indexing Methods for Temporal Relation. IEEETransactions on Knowledge and Data Engineering, 1993, 5(3): 496～509

8 Vram K. Temporal databases: Access structures, search methods, migration strategies, and declustering techniques: [Ph D dissertation]. Arlington: The University of Texas, 1994