

DBSCAN 在非空间属性处理上的扩展

孙志伟, 赵 政

(天津大学 电子信息工程学院 天津 300072)

(szw_1013@sina.com)

摘 要:在很多有效的聚类算法中,DBSCAN 算法对于聚类空间数据有着非常好的性能,依赖于基于密度的聚类定义,DBSCAN 可以发现任意形状的聚类,而且执行效率很高。但是,DBSCAN 没有考虑非空间属性,而非空间属性对聚类的结果也起着十分重要的作用。在 DBSCAN 的基础上,参考 DBRS 的概念,进一步考虑了非空间属性的数据类型,从而提出了可以处理空间和非空间数据的新的聚类方法,并给出了主要的算法。

关键词:空间数据挖掘;空间聚类;非空间属性;密度

中图分类号:TP311.131 **文献标识码:**A

Extension of DBSCAN with non-Spatial attributes

SUN Zhi-wei, ZHAO Zheng

(School of Electronic & Information Engineering, Tianjin University, Tianjin 300072, China)

Abstract: In many effective algorithms for cluster, DBSCAN algorithm is outstanding for its good performance in spatial data. Relying on a density-based notion of clusters, DBSCAN can discover clusters of arbitrary shape. But it can't support non-spatial attributes, in some application of cluster, the non-spatial attributes play important role. Based on the DBSCAN, referencing some notion of DBRS and considering data type of non-spatial attribute, the paper proposed a method of extension of DBSCAN and gave main algorithm. The algorithm can operate spatial and non-spatial attribute.

Key words: spatial data mining; spatial cluster; non-Spatial attribute; density

聚类的任务是把一个数据集分解或划分成组,使同一组中的点彼此相似,但与其他组中的点尽可能不同。聚类属于无指导的学习。目前已经提出了许多有效的聚类算法,如 CLARANS^[1]、BIRCH、DBSCAN^[2]、STING 以及 WaveCluster 等算法。基于 DBSCAN 算法,本文提出了一种同时考虑数据的空间和非空间属性的方法,通过在聚类时分别处理空间和非空间属性,形成各自的聚类,然后对两种聚类结果进行笛卡尔积,从而得到实际数据的聚类结果。

1 聚类的介绍

目前对大型数据库提出了很多有效的算法,这些算法主要分为以下几类:分区方法、层次方法、基于密度的方法、基于网格的方法和基于模型的方法。Ng 和 Han 提出的 CLARANS 把采样技术 PAM 结合起来,是第一个把聚类技术引入空间数据聚类分析的方法。

分层方法是对给定数据对象集合进行层次的分解,使用系统树图表示,根据层次的分解的形成,又可分为凝聚法和分裂法。但这种方法对发现球型聚类比较适合,不能发现任意形状的聚类,因此我们提出了基于密度的聚类方法,主要思想是:只要临近区域的密度(对象或数据点的数目),达到了某个标准,就进行聚类。基于网格的方法把对象空间量化为有限数目的单元,形成了一个网格结构,所有聚类都在网格结构上进行。基于模型的方法为每个簇假定了一个模型,寻找数据对给定模型的最佳拟合。这种方法试图优化给定的数据和某些数学模型之间的适应性。

在处理空间属性时大部分算法都只考虑空间属性,而不考虑非空间属性,基于随机搜索的 CLARANS 算法在文献[1]中给出了两种空间数据挖掘算法:SD (CLARANS) 和 NSD (CLARANS)。这两种算法都需要借助于一个工具 DBLEARN,算法实际上是 DELEARN 和 CLARANS 的合并处理。

另外在文献[5]提出了可以处理空间和非空间属性的算法 DBRS,它是在 DBSCAN 基于密度的概念的基础上,对非空间属性提出了一个纯度的概念。纯度是邻域内相同非空间属性的点数和邻域内所有点数的比值。如果这个比值大于指定的最小纯度则成为匹配邻域,查询点成为核心点。则开始聚类。由于使用了随机样例的方法,可以发现近似聚类,所以效率较高。这种方法的缺点是对非空间数据的处理,按照匹配邻域^[5]的定义,是按照属性的绝对相同来处理的,但是很多时候对于数值型数据来说,同一类型的对象的数值型属性是在一个区间内的不固定的值,这样居于属性完全相等的纯度概念并不完全适合,因此基于纯度的概念将无法满足不同情况。在这 DBRS 总空间和非空间属性的半径采用了同一个参数,而实际情况可能不同,因此最好两个半径能够有所不同,以满足变化的需要。

在文献[3]、[6]中提出的 ROCK 和 CATCUS 是聚类的分类方法,单独考虑非空间属性的分类聚类。文献[4]中介绍的 GDBSCAN 提出的空间和非空间属性的算法,并提出了邻居 NPred 和集的加权的势的概念,势(wCard)由用户定义的函数表示,例如使用非空间属性数据的和,由用户根据需要来

指定,并指定一个最小的阈值 *MinWeight*,但是同样只能发现符合指定具体条件的聚类,不能够发现空间中所有的自然聚类。

2 DBSCAN 在非空间数据上的扩展

DBSCAN 是基于密度的方法,可以发现任意形状的聚类,关键思想是对聚类中的每一个对象,在给定的最大半径 *Eps* 所形成的邻域内 *NEps(p)* 最少要包含 *MinPts* 个数据点。发现聚类主要是基于这样的事实:聚类由它的多个核心对象所决定。采用索引的情况下,计算的复杂度是 $O(n \log n)$, 否则是 $O(n^2)$ 。同时算法对用户定义的参数很敏感。

2.1 基本思想

处理非空间数据,首先需要分清非空间数据的类型。非空间数据主要的数据类型包括整型、浮点型、字符型、逻辑型、日期型等几种数据类型,每种数据在数据库中都有和它们对应的类型,主要分为两类数据:数值类和非数值类数据。前者是定量的、连续的,后者是定性的、非连续的。数值类可进行加减运算,日期型两数据间的距离可以使用两个时间点的差来表示,此时的密度表示的是时间点的稠密程度。逻辑型数据则属于字符类。

数值类数据的聚类实际上是一维空间中的基于密度的聚类:在指定的一维距离范围 *Eps* 内符合最少的数据值的个数 *MinStand*。

而对于非数值类数据来说,要表达聚类的意义有三种标准:一是在某个邻域内的相同非空间属性绝对的数量;二是在某个邻域的相同非空间属性相对的数量,即要求在指定空间半径 *Eps* 内的相同属性的数据的数据的数量与总数量相比 *Ratio*: $\left| \{q \in D \mid \text{dist}(p, q) \leq Eps \text{ and } p.\text{prop} = q.\text{prop}\} \right| / \left| \{q \in D \mid \text{dist}(p, q) \leq Eps\} \right|$ 要不小于一定的域值 *MinStand*。满足这样条件的区域成为匹配邻域。在后面的算法中给出了两种标准对 DBSCAN 的修改要求。另外还用一种可能就是采用字符数据的每一个值就表示一个聚类,这实际上是一种数据分类^[3,6]。用户在实际应用中可以根据需要分别使用三种标准或进行比较相应的结果。第三种标准目前提出的算法效率都很高,基本是线性的。相对于 DBSCAN 来说,我们主要以前两种为主来介绍。

2.2 算法描述

数据结构说明:表示聚类 *Id* 的数组 *Cluid*[], 分别表示最终聚类 *id*, 空间聚类 *Id*, 非空间聚类 *Id*; 表示空间位置的 *X, Y*; 表示非空间属性值和类型 *NSAttrValue*[], *NSAttrType*[]。其中 *NSAttrType* 表示对不同的类型有不一样的条件处理或者进行不同的运算(数值型或日期型)作用。

聚类时空间与非空间的半径要求可能不同,对于数据的输入参数标准也使用数组。为了空间和非空间属性的统一,表示半径仍然使用 *Eps*[], 而另外一个参数使用 *MinStan*[], 但每个元素表示的意义可能不同,如 *MinStan* 可能是一个绝对的数量值或者是一个比值。这样,用户可以根据情况设定参数找到最合适的聚类。同时采用数组可以方便的把算法应用到多个非空间属性的聚类问题上,避免使用单一变量的限制。算法是对 DBSCAN 的一个改进版,顺序处理每个属性,最终的聚类结果和空间数据和非空间数据的处理顺序没有关系。下面给出相应的算法描述:

DBSCAN_Ext(SetOfPoints, Eps, MinStand)

For i: = 0 to NsAttrNum do

CreateCluster(SetofPoints, eps[i], MinStan[i], i);

//分别聚类空间和非空间数据

MarkEndClu(SetofPoints);

//标记最终聚类结果

对空间和非空间属性的每一项进行聚类,其中 *snflag* 表示空间和非空间的标志,0 表示空间,大于 0 表示非空间。下面是对 DBScan 的修改,在输入参数上增加了 *snflag*。

CreateCluster(setofpoints, eps, MinStan, snflag)

Cluid = this. NextId (NOISE);

for(int i = 0; i < setofpoints. count; i++) do {

Point = setofPoints. get(i);

if Point. Cluid[snflag + 1] == UNCLASSIFIED) then {

if (ExpandCluster(setofpoints , i, Cluid, eps, MinStan, snflag))

then {

Cluid = NextId (Cluid);

}

}

ExpandCluster 是最重要的一个函数,在其中需要增加 *snflag*, 还应用于改变点的 *Clu* 数组的下标表示,同时还要判断当前处理的非空间属性的数据类型,因为类型不同,可能判断的条件有所不同,下面算法中给出了数值类和字符类数据的不同条件的处理。

ExpandCluster(SetOfPoints, Point, Clusterid, Eps, MinStan, snflag):

boolean

seeds = RangeQuery(Point, Eps, snflag);

if (snflag >= 1) then {

if NSAttrType[snflag - 1] in NumType then

condition = seeds. Count < MinStan;

else {

condition = seeds. Ratio < MinStan; }

}

else {

condition = seeds. Count < MinStan; }

if (condition) then {

ChangeCluID(Point, NOISE, snflag);

return false; }

...

return true;

省略的部分可以参照 DBSCAN 和已列出算法作相应修改即可;在算法中需要注意的是距离 $\text{dist}(p, q)$ 的变化,即在判断是否是邻域内点时的距离函数 *GetDist* 中对于空间度量使用欧几里德距离,而非空间属性中使用两点的属性的差的绝对值即: $\text{dist}(p, q) = |p.\text{prop} - q.\text{prop}|$, 属性可以是任何可以做减法运算的类型。在使用时注意实际非空间属性的单位表示,以决定参数的选择。前面提到的字符类数据聚类的三种标准,上面的算法是针对第二种标准给出的,第一种标准的修改相对简单一些,对于第三种标准可以参考文献[3]、[6]的方法,聚类的速度是线性的。

函数 *MarkEndClu*: 空间和非空间聚类的综合,即聚类的最终标记,进行每种属性聚类结果的笛卡尔积。显示时应附加说明每种聚类的含义。

MarkEndClu(setofpoints)

Cluid = NOISE;

clu[Cluid, 0] = Cluid; 其他列都为 NOISE

```

for i from to setofpoints.size do {
    find = false; Point = setofPoint.get(i);
    for j=0 to CluId do {
        if Point 的所有空间和非空间聚类 Id 与 clu 的非 0 列元
            组相等 then{
                setofpoints[i].CluId[0] = clu[j,0];
                find = true; }
    }
    if (!find) then {
        CluId 自增一;
        把新的匹配写进 clu 数组
        setofpoints[i].CluId[0] = CluId;
        find = false; }
}

```

在实际使用中,可以对有多个非空间属性操作时的非空间属性采用数组表示,同时说明各类属性值和类型可以各使用数组表示,这样在处理时判断相应属性类型,从而达到把非空间属性中各种属性值的数据合并到一起。

3 性能分析

试验采用的是文献[1]中表示房屋数据的合成数据集,如图1所示,既包含数值类数据(房屋大小和价格),也包含字符类数据(房屋类型)。由于数据集的数据分布特性,采用相对和绝对标准度量的聚类的结果基本相同,图2显示的对空间属性和房屋类型(第二种标准)的组合聚类结果(参数:空间属性 $Eps = 30, MinPts = 20$; 房屋类型 $Eps = 10; MinPts = 0.5$),整体是空间聚类,其中实心圆点表示 mansion,而十字表示是 single-house。结果显示算法对聚类空间和非空间属性的数据是有效的。

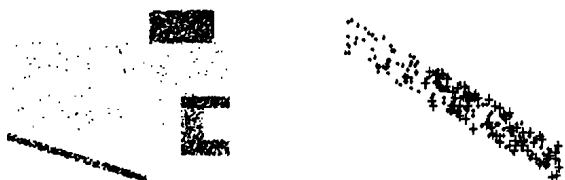


图1 2.5k个房屋的空间分布 图2 左下角数据的房屋类型聚类

假设数据库中对象的数目是 n , 空间聚类的数量是 c_0 , 非空间属性的聚类的数量是 $c_i, i \in [1, m], m$ 是非空间属性的数量, 考虑使用空间索引和 B 索引, 则空间聚类的计算复杂度是 $O(n \log n)$; 非空间数据中每一种属性复杂度和处理空间数据的复杂度是相同的, 即 $O(n \log n)$ (第三种标准的复杂度是 $O(2n)$); 在最后把空间聚类与非空间聚类合并时, 最终的聚

类数量最大值是 $c = \prod_{i=0}^m c_i$, 复杂度最大是 $O(nc)$ 。总的计算复杂度是 $O(n((m+1)\log n + c))$ (一般来说 $m, c \ll n$)。相比来说算法的计算复杂度较大, 但是其他算法只能找到符合指定要求的聚类, 而此方法可以找到所有自然存在的聚类, 稍加扩展可得到属性子空间的聚类结果。

4 结语

本文在 DBSCAN 的基础上扩展了对该算法处理非空间属性的能力, 对非空间属性的处理要准确的聚类, 应该要区分非空间属性数据的类型, 而本文提出的改进在数值类数据上直接使用在一维数据上的距离度量的, 在字符类数据上则指出了不同的聚类标准。试验表明算法能够很好的完成聚类的结果。下一步的工作是进一步合理化核心点的采样工作以减少算法运行时间。使随着数据量的增加尽可能的实现线性增长, 而且 DBSCAN 一些扩展使用的采样技术并不完全适合于非空间数据, 如何在大数据库下提高速度需要进一步的考虑。另外, 是否能够把空间属性和非空间属性以及非空间属性的数值类和非数值类属行的进一步的结合, 即寻找一种能够表示相应属性的相似度和相异度的度量也是研究的方向。

参考文献:

- [1] NG RT, HAN J. Efficient and Effective Clustering Method for Spatial Data Mining [A]. Proceedings of the 20th Int'l Conf. on Very Large Data Bases [C]. San Francisco, Morgan Kaufmann Publishers, 1994. 144 - 155.
- [2] ESER M, KEIEGEL H - P, SANDER J, et al. A Density Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise [A]. Proceedings of KDD'96 [C]. Portland, AAAI Press, 1996. 226 - 231.
- [3] GANTI V, GEHRKE J, RAMAKRISHNAN R. CACTUS-Clustering Categorical Data Using summaries [A]. Proceedings of ACM SIGKDD, International Conference on Knowledge Discovery & Data Mining [C]. San Diego, CA USA, 1999. 73 - 83.
- [4] SANDER J, ESTER M, KRIEGL H-P, et al. Density - Based Clustering in Spatial Databases: The Algorithm GDBSCAN and Its Applications [J]. Data Mining and Knowledge Discovery, 1998, 2 (2): 169 - 194.
- [5] WANG X, HAMILTON HJ. DBRS: A Density - Based Spatial Clustering Method with Random Sampling [A]. Proceedings of the 7th PAKDD [C]. Seoul, Korea, 2003. 563 - 575.
- [6] GUHA S, RASTOGI R, SHIM K. Rock: A robust clustering algorithm for categorical attributes [A]. Proceedings of the 15th International Conference on Data Engineering [C]. Sydney, Australia, 1999. 512 - 521.

(上接第 1378 页)

参考文献:

- [3] ARAWAL R, et al. Paralled Mining of Association Rules [A]. IEEE Transactions on Knowledge and Data Engineering [C], 1996, 8(6): 962 - 969.
- [4] PARK JS, et al. Efficient Parallel Data Mining for Association Rules [A]. Proceedings of the 4th International Conference on Information and Knowledge Management [C]. Baltimore, 1995. 31 - 36.
- [5] CHEUNG DW, et al. Efficient Mining Of Association Rules In Distributed Databases [J]. IEEE Transaction On Knowledge And Data Engineering, 1996, 8(6): 910 - 953.
- [6] CHEUNG DW, et al. A Fast Distributed Algorithm for Mining Association Rules [A]. Proceedings of the International Conference on Parallel and Distributed Information Systems [C]. Miami Beach, Florida, USA, 1996. 31 - 42.
- [7] AGRAWAL R, SRIKANT R. Fast Algorithms for Mining Association Rules [A]. Proceedings of the 20th International Conference on Very Large Databases [C]. Santiago, Chile, 1994. 487 - 499.
- [8] 冯玉才, 冯剑林. 关联规则的增量式更新算法 [J]. 软件学报, 1998, 9(4): 301 - 306.