一种基于密度的不确定数据离群点检测算法

姜元凯 郑洪源 丁秋林

(南京航空航天大学计算机科学与技术学院 南京 210016)

摘 要 不确定数据普遍存在于如移动计算、RFID技术和传感器网络等大量应用之中。由于不确定数据的离群点检测算法可以提高服务质量,提出一种基于密度的不确定数据离群检测算法RLOF。该算法引入一种R2-tree结构,有效降低了计算局部离群因子时的时间复杂度,同时降低了不确定数据集中的数据更新成本以及海量数据维护成本。理论分析和实验结果充分证明了该算法是有效可行的。

关键词 不确定数据,离群点检测,R2-tree索引,最小充分邻域

中图法分类号 TP274 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.4.034

On Density Based Outlier Detection for Uncertain Data

JIANG Yuan-kai ZHENG Hong-yuan DING Qiu-lin

(College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

Abstract Uncertain data generally exist in a large number of applications, such as mobile computing, sensor networks and RFID technology. Outliers detection algorithm can improve the quality of these services. An uncertain data outlier detection algorithm based on density RLOF was proposed. This algorithm introduces a R2-tree structure, which effectively reduces the time complexity when calculating local outlier factor. It also reduces the cost of data updating in the uncertain data set and the maintenance cost of a massive data. The theoretical analysis and experimental results fully prove that the algorithm is effective and feasible.

Keywords Uncertain data, Outlier detection, R2-tree index, Minimal sufficient neighborhood

1 引言

离群点检测问题是数据挖掘技术的重要研究领域之一,其任务是从大量复杂的数据集中发现小部分异常的、与常规数据模式显著不同的新的数据模式。不确定数据广泛存在于文本分析、信息检索和社交网络等各种应用之中。产生不确定数据的原因有多种,如数据噪声、数据丢失、传输延迟,以及测量不精确或不完整等。另外,大规模应用如传感器网络和RFID技术更是产生了大量的不确定数据。如果不对相应的不确定数据进行处理,会严重影响到应用程序的服务质量。

离群点检测算法可以分为以下几类:

1)基于统计的离群点检测。基于统计的离群点检测方法是一种基于模型的方法,也就是为数据集创建一个模型,并且根据数据对象拟合模型的情况来评价它们。大多数基于统计的方法都是通过构建一个概率分布模型,来考虑数据对象符合这个模型的程度。

2)基于距离的离群点检测。基于距离的离群点检测方法是将到其它数据对象的距离大于等于某设定阈值的数据对象判定为离群点。

3)基于密度的局部离群点检测。Breunig等人[1]提出了局部离群因子的概念LOF,这是一种基于密度的方法。通过数据空间的所有维度来计算对象的距离,进而计算对象的可达密度,最后通过局部的离群度来判断离群点。LOF算法成功地区别了不同群组之间的密度差异。

4)基于聚类的离群点检测。基于聚类的方法将数据集分成若干个簇,不属于任何簇的对象就是离群点,经典的算法有CLIQUE、BRITH、STING、DBSCAN及模糊k均值聚类方法等。

目前对不确定数据的离群点检测研究主要有以下两篇文献:Yu和Aggarwal在文献[6]中指出,在处理多维数据时,即使数据对象不是离群点,它属性的不确定级别也直接导致它会被视作离群点。由此提出了基于遍历子空间和密度估计的不确定数据离群点检测算法。算法通过定义平概率来量化不确定数据对象在一个密集区域出现的概率,将低于特定阈值的数据对象定为离群点。而文献[13]研究了元组级不确定数据集中基于距离的不确定离群点检测,利用非离群点的过滤性质来减少检测次数,从而提高算法效率。

本文借鉴LOF算法的特点,提出一种适用于不确定数据的基于密度的局部离群点检测算法RLOF。该方法通过引入R2-tree数据结构,在提高算法效率的同时,能够实时地响应数据集的更新操作。R2-tree属于R树家族,能够同时满足空间、海量数据的需求。

到稿日期:2014-05-20 返修日期:2014-08-07 本文受江苏省产学研联合创新资金项目(SBY201320423)资助。

姜元凯(1990-),男,硕士生,主要研究方向为知识工程、数据挖掘、人机交互,E-mail:972596721@qq.com;**郑洪源**(1973-),男,博士,副教授,硕士生导师,主要研究方向为知识工程、信息系统与信息安全、人机交互;**丁秋林**(1936-),男,博士,教授,博士生导师,主要研究方向为信息系统、企业信息化。

2 不确定数据离群点检测 RLOF 算法设计

2.1 RLOF 算法的相关概念

以下是 RLOF 算法的几个基本定义。

定义 1(不确定数据对象 o 及其邻居组成的可能世界的概率) 不确定数据库 $D_u=\{X_1,\cdots,X_n\}$,每个元组 X_i 由两部分组成,$\overline{X}_i$ 表示数据记录,$\psi(\overline{X}_i)$表示该数据记录的概率值(或可信度)。假设各元组之间相互独立,$N(o)=\{w|w\subseteq D_u, o\in w\}$表示一个由对象 o 和它的邻居组成的可能世界实例,则该实例的概率由式(1)计算得到:

$$prob_{N_{(o)}}=\prod_{X_i\in N(o)}\psi(X_i)\prod_{X_i\notin N(o),X_i\in D_u}(1-\psi(X_i)) \tag{1}$$

但该概率值随着数据量的增加而越来越小,不利于阈值设置。我们用可能世界实例中各数据对象的概率均值来代表该实例的概率,记作式(2):

$$prob_{N(o)}=\frac{\sum_{X_i\in N(o)}\psi(X_i)}{|N(o)|} \tag{2}$$

定义 2(数据邻域的平均半径) 一个可能世界实例中除去的数据对象 o 的数据点构成了对象 o 的邻域 $N(o)$,邻域中任意数据对象 p 到 o 的距离记作 $d(o,p)$;该邻域中的所有数据对象到对象 o 的平均距离定义为式(3):

$$R_{avg}(N_{(o)})=\frac{\sum_{p\in N(o)}d(o,p)}{|N_{(o)}|} \tag{3}$$

定义 3(数据对象 o 满足邻居数量为 k 且概率大于阈值 θ 的最小邻域) 在 LOF 算法中,Breuning 使用数量阈值来计算局部离群因子,但考虑到不确定数据的概率属性,概率阈值不可避免地成为了标识局部离群点因子的一个重要指标。设 k 为一自然数,θ 表示可能世界实例的概率阈值。数据对象 o 可以和它的邻居数据对象构成众多的可能世界实例。

对于满足 1)$N_{\min}(o)$;2)$prob_{N_{(o)}}\geqslant\theta$ 的众多可能世界实例 $N_i(o)$,拥有最小 $R_{avg}(N_{(o)})$ 的邻域 $N_{(o)}$ 称为 o 的最小邻域,最小邻域中到数据对象 o 的最大距离称为最小邻域的核心距离,记作 k,θ-$\phi_{\min}(o)$,简写为 $\phi_{\min}(o)$。

定义 4(不确定数据对象 o 的最小充分邻域) 数据对象 o 的最小充分邻域则是所有到 o 的距离小于等于核心距离的点的集合,记作式(4):

$$N_{k,\theta\min}(o)=\{q\in D_u\backslash o|d(o,q)\leqslant\phi_{\min}(o)\} \tag{4}$$

本文中将 $N^{(o)}_{k,\theta\min}$ 简写为 $N_{\min}(o)$。

由于不同局部空间内的数据分布密度不同,因此对于密度较大的区域,邻域的核心距离较小;而密度较小的区域,该数值则较大。同样地,对于可信度较高的区域,核心距离较小;反之,较大。一般情况下,一个这样的邻域充分包含所有满足条件的数据点,其与定义 3 中由可能世界实例构成的邻域的区别在于包含了其他因概率较小而在选择最小邻域时被过滤掉的点,因此集合中至少包含了 k 个数据对象。

定义 5(不确定数据对象的局部可达密度) 与 LOF 算法类似,首先定义 p 关于数据点 o 的可达距离,如式(5):

$$reachability\text{-}dis_{k,\theta\min}(o,p)=\max\{\phi_{\min}(p),d(o,p)\} \tag{5}$$

我们通过可达距离可以在局部区域内区分出离群点。若数据对象 p 为离群点,则可达距离有很大可能取 $d(o,p)$,反之取 $\phi_{\min}(p)$。

局部可达密度可以定义为式(6):

$$lrd_{k,\theta\min}(o)=1/\left[\frac{\sum_{p\in N_{\min}(o)}reachability\text{-}dis_{k,\theta\min}(o,p)}{|N_{\min}(o)|}\right] \tag{6}$$

通过分析可以看出,如果数据对象 o 的偏离程度较大,则其到邻域内其他对象的可达距离较大,所以使得 $lrd_{k,\theta\min}(o)$ 较小。而处于同一聚类群组中的数据对象的值相对接近,与偏离较大的数据点之间形成较大差异。

定义 6(不确定数据对象的局部离群因子) 局部离群因子通过式(7)计算:

$$RLOF_{k,\theta}(o)=\frac{\sum_{p\in N_{\min}(o)}\frac{lrd_{k,\theta\min}(p)}{lrd_{k,\theta\min}(o)}}{|N_{\min}(o)|} \tag{7}$$

该公式计算出数据对象 o 的最小充分邻域中所有数据对象的局部区域的平均值,一个离群点的 $lrd_{k,\theta\min}(o)$值较小,所以 $RLOF_{k,\theta}(o)$较大。由此成功区分了不确定数据的不同密度。

2.2 RLOF 算法的复杂度分析

计算不确定数据对象的局部离群因子,RLOF 算法可以大致分为 3 个步骤:第一步,确定数据对象的最小充分邻域;第二步,计算数据对象的局部可达密度;第三步,计算数据对象的局部离群因子。

我们发现在计算对象 o 的局部离群因子时,除了要计算对象 o 的 lrd 值,同样需要计算 o 的邻居对象 p 的 lrd 值。如果 p 属于 m 个数据对象的最小充分邻域,难免会被重复计算 m 次最小充分邻域和对象间的核心距离。其时间消耗随着数据集的增加和阈值的提高而迅速增长。因此,可以考虑增加空间开销来降低算法时间复杂度。在算法执行之前,预先计算所有数据对象的最小充分邻域,将核心距离 $\phi_{\min}(o)$和邻居对象集合 $N_{\min}(o)$保存以供后续查询。因此,需要增加 N^2+N 的空间开销。实验证明,该改进[11]极大地提高了算法执行的时间效率,效率的提升随着计算规模的扩大而增大。

在计算数据对象局部离群因子的 3 个步骤中,第一步的时间复杂度的阶最高,消耗的时间最长,降低其时间复杂度的阶成为提高 RLOF 算法效率的核心问题。由此可以利用索引来提高最小充分邻域的计算效率。同时还需要对各对象预先计算的最小充分邻域和核心距离进行管理。因此,下文提出一种基于 R2-tree 树的改进来提高 RLOF 算法的执行效率。

2.3 基于 R2-tree 树的算法改进

空间索引是对存储在介质上的数据位置信息的描述,用来提高系统对数据获取的效率。当各种海量复杂数据存储于外存时,如果对磁盘上的数据的位置不加以记录和组织,每查询一个数据项都要扫描整个数据文件,则这种访问磁盘的代价将严重影响系统的效率。数据多维性使得传统的 B 树索引不再适合,因为 B 树所针对的字符、数字等传统数据类型是在一个良序集之中,即都是在一个维度上。Guttman[14] 及后人提出的改进形成了一个繁荣的 R 树索引族,是目前所流行的空间索引。

在确立最小充分邻域时,我们希望尽可能少地访问数据对象,以减少 I/O 消耗。R 树索引提供了数据对象间的空间关系,可以利用这种特性来完成最小充分邻域的查询。另一个问题是,当数据集发生变化时,无法实时地、以较小的代价维护最小充分邻域。为高效地确认和更新数据对象的最小充分邻域,本节提出了一种基于 R 树的双索引数据结构 R2-tree。

2.3.1 R2-tree 数据结构

我们用每个数据对象的多维属性来描述其空间位置信息,利用欧氏距离公式计算各数据对象间的关系。除此以外,

每个数据对象还具有一个描述其最小充分邻域的属性。两者相互独立，为避免建立 2 个不同索引，将其合二为一，提出了 R2-tree 数据结构。

与传统的 R 树相同，R2-tree 也分为叶子节点和非叶子节点。如图 1 所示，叶子节点的数据结构为：

〈Value，Radius，Parent，Neighbor〉

其中，Value 指向数据对象的 d 维属性和概率属性；Rad- ius 为数据对象的核心距离；Parent 指向该节点的父节点；Neighbor 指向该数据对象的邻居数据对象的列表。非叶子节点的数据结构为：

〈Node[]，Rect，Max_Radius，Parent〉

其中，Node 指向各子节点；Rect 为包含其所有子节点的最小边界矩形；Max_Radius 为所有子节点中核心距离的最大值；Parent 指向该节点的父节点。

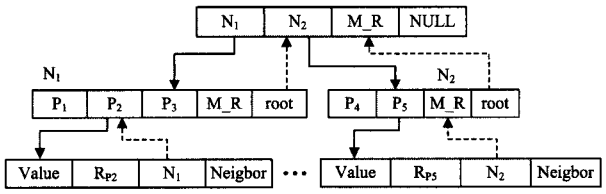

图 1　R2-tree 的数据结构

生成一棵 R2-tree 主要分为两个步骤：第一步，初始化一棵 R2-tree，将数据对象根据其空间属性插入到各个节点中，设置各个节点的 Value、Parent、Rect 属性，其插入、分裂方式与 R 树相同，这里就不再详述；第二步，计算每个数据对象的最小充分邻域，利用 Parent 指针由底至顶填充 Radius 和 Max_Radius 属性。

2.3.2　基于 R2-tree 的最小充分邻域的确定

根据 R2-tree 节点中的 Node 和 Rect 属性，当需要进行一个空间查询时，我们只需要从根节点开始，由顶至底访问少数几个叶子结点的最小边界矩形，查找出满足条件的数据对象。

算法 1　最小充分邻域的确定算法

输入：查询对象点 q，根据数据集 D 建立的 R2-tree 树，每个先序队列 queue 节点的数据结构为〈node，key〉，变量 k，概率阈值 θ

输出：对象点 q 的最小充分邻域

1)初始化队列 queue，R2-tree 树的根节点入队：Enqueue(queue，root-Node，0)，初始化变量 i=0，prob=1，N[k]，No[]；

2)While(队列非空) do

执行出队：e←Dequeue(queue)

If(e 是数据点或它的边界矩形) then

If(e 是数据对象的边界矩形 and 队列非空 and 该对象到 q 的距离＞队列头到 q 的距离)

Enqueue(queue，object，DIST(object，q))

else

If (i<k) then

N[i]=e

将 e 输出到 No

i++

Elseif (i==k)

对于 N 中的 k 个数据点，计算其可能世界模型概率，赋值为 prob

If (prob≥θ) then

判断 DIST(q，e)是否等于 No 中点到 q 的距离，若是则将 e 输出到 No

Else

删除 N[]中概率值最小的点

i=i−1

End If

End If

End if

ElseIf (e 是一个叶子节点) then

For each Object in 叶子节点 e do

If DIST(q，object)≥DIST(q，e) then

Enqueue(queue，object，DIST(q，object))

End if

End do

Else /* e 是非叶子节点

For each e 的子节点 child do

Enqueue(queue，child，dist(q，child))

End do

End If

End Do

3)输出 No[]中的所有数据点

本算法采用一个优先队列保存 R 树中的节点，将 queue 初始化为根节点，设置两个数组，一个保存最小邻域，其大小恒为 k；另一个作为输出，保存最小充分邻域。每次循环读取队列中队首元素，以确保每次获得的节点到查询对象的距离最小，将它的孩子节点根据 MINDIST 顺序添加到 queue 中，不断执行该过程直到队首元素为数据对象，在该过程中始终保持队首元素到查询距离最短。我们设置一个变量 i 用来记录当前选中存入 N 数组的数据对象个数，当满足数量阈值 k 时，计算其可能世界模型发生的概率。如果 prob 不满足概率阈值，则删除 N 数组中概率最小的数据对象，这样在进行下一步循环的时候，会有新的数据对象替换进来，直至满足概率阈值。当 prob 满足概率阈值时，得到的输出并不是最小充分邻域，需要将到查询点距离等于最小邻域半径的数据对象也添加到输出数组 No 中，这样最后得到的就是查询点 q 的最小充分邻域。

在得到数据对象的核心距离后，从叶子节点开始向上利用 Parent 指针更新各父节点的 Radius 和 Max_Radius 属性值。

2.3.3　最小充分邻域的动态更新

数据集 D 常常会进行更新操作。然而新加入的节点或者删除的节点对于其它数据对象的局部离群因子的影响往往是局限在某个区域的，如果按照原算法，则需要对新数据集中所有数据对象进行重新计算，但代价太大。所以通过 R2-tree 找到受影响的对象，并且对其最小充分邻域和局部离群因子进行更新是本节的主要内容。

定理 1　如果数据对象 q 到 R2-tree 中任意节点的 rect 的最小距离大于该节点的 Max_Radius，则该节点中的数据对象可以被排除。

Max_Radius 为该节点中所有数据对象的核心距离的最大值。数据对象 q 到 Rect 的最小距离即到 Rect 中数据对象的最小距离如果大于 Max_Radius，则 q 一定不属于 rect 里任何数据对象的最小充分邻域，所以 rect 中的数据对象可以被排除。

根据定理 1，可以得到更新数据对象的查询算法。

算法 2　ObjectSearch(Node node，Object q)

输入：R2_tree 的节点 n，查询数据对象 q

输出：需要更新的数据对象

1)如果 n 是一个叶子节点，n=〈value，radius〉，如果数据对象 q 到该

节点指向的数据对象 value 的距离小于 radius，则输出该点；

2)如果 n 是一个非叶节点，对于它所指向的每个节点 B=〈Node，Rect，Max_Radius〉，如果 q 到 node 的最小距离大于等于 Max_Radius，则递归 ObjectSearch(Node，q)。

3 实验结果及分析

本节通过实验对 RLOF 算法的效率和检测精确度进行分析验证。实验的硬件环境是：Inter CPU 2.93GHz，内存 2GB。软件环境是：操作系统为 Microsoft Windows7 Ultimate，实验程序使用 C++编写，开发环境为 Microsoft Visual Studio 2010。测试所用的数据为模拟数据集 TestDS 和数据集 KDD CUP1999，从中分别选取 2～25 个数值型属性转化为相应的不确定数据集。

1)R2-tree 索引的性能测试

在 Guttman 提供的 R-tree 代码的基础上实现了基于 R2-tree 的索引。通过不同规模、不同维度的数据集和不同数量、概率阈值对最小充分邻域的确定算法的测试，得出了在 R2-tree 索引条件下该算法的执行效率。由 2.3 节可以发现，R2-tree 索引首先需要由顶至底建立一棵 R-tree，由此可以查询得到数据对象的最小充分邻域和邻域的核心距离，再由底至顶填充 radius 属性，构成一棵完整的 R2-tree。表 1 是在采用 R2-tree 索引后，数据维度为 2 时，针对不同规模数据集和不同阈值，算法的实际运行时间。其中将 R2-tree 的节点分支树设为 8，查询时间包括 I/O 和 CPU 时间。

表 1 各个过程实际运行时间(ms)

数据规模	建树时间	查询时间 (k=10,20,50)			填充时间
10000	165	2	3	7	<1
20000	355	2	4	8	<1
50000	982	3	4	8	1
100000	1994	3	4	8	1
200000	4102	3	4	9	1

分析实验结果可以看到，随着数据规模的增加，只有建树时间随之增长；而最小充分邻域的确认时间和填充时间却增加缓慢，原因是其时间消耗只与 R 树的层数相关，而层数的增长远远慢于数据规模的增长；查询时间也会随着阈值的提高而增加。

考虑到 2.2 节中提出的计算所有数据对象的最小充分邻域以备用，那么本文算法消耗的总时间可以用如下公式计算：

总时间＝建树时间＋数据规模＊(查询时间＋填充时间)

其中，查询时间为确定单个数据对象最小充分邻域的时间，填充时间为自底向上填充 radius 属性的时间。图 2 所示为不同规模数据集和不同阈值时，建立 R2-tree 树的时间消耗。在此之后对数据集进行查询、添加、删除操作即可在极短的时间内完成。

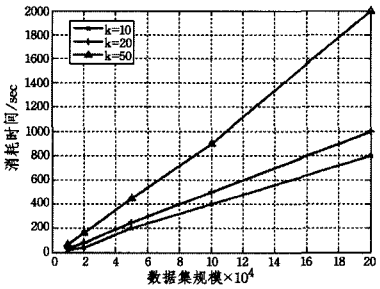

图 2 建立 R2-tree 的时间消耗

接下来考虑多维数据对象的情况，假设数据规模为 10000，表 2 给出了不同维度下的 R2-tree 的执行情况。

表 2 不同维度下的实际运行时间(ms)

数据维度	建树时间	查询时间 (k=10,20,30)			填充时间
2	165	2	3	7	<1
3	173	3	5	10	1
5	243	6	9	17	1

图 3 为不同维度建立 R2-tree 的时间消耗。

可以看出，建树时间和查询时间都会随着维度的增加而增加，但是由表 1 考虑到查询时间随数据规模的增长变化缓慢，建立 R2-tree 在总时间上仍是平缓上升，时间复杂度为 $O(kn)$，属于可控范围。

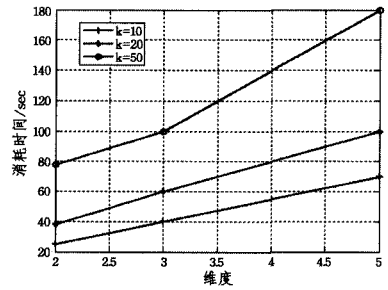

图 3 不同维度建立 R2-tree 的时间消耗

2)RLOF 算法的精确度

为了测试 RLOF 算法的精确度，使用 TestDS 数据集。假定真实的离群点在不确定数据集中占有 5%，其每条数据对象的概率值满足正态分布。可以通过将检测出来的离群点和真实的离群点进行比较来评价一个离群点检测算法的好坏，所占比例越高说明该方法的性能越好：

$$\text{精确度}=\frac{\text{正确找到的离群点数}}{\text{离群点总数}}$$

为充分考虑数量阈值与概率阈值对于不确定离群点检测的影响，我们选取不同的数量阈值和概率阈值来进行实验，实验结果如表 3 所列。

表 3 不同阈值对于 RLOF 检测精确度的影响

概率\数量阈值	K=5	K=20	K=50
θ=0.7	0.86	0.91	0.93
θ=0.75	0.88	0.92	0.93
θ=0.8	0.89	0.92	0.93

从中可以发现，随着阈值的提高，最小充分邻域扩大，各数据对象的局部离群因子更加准确，使得检测精度提高。但是，其检测精度最终趋向于一个稳定值。考虑到较高的阈值会增加算法 1 的时间开销，我们在选取数量阈值和概率阈值时，并不是越高越好，而应考虑各个因素，选取折中方案。

3)同类算法比较

由上文可知，RLOF 采用局部离群因子来表征数据对象的孤立程度，该值取决于其数据对象邻域内所有数据的局部可达密度，是一个相对属性，因而可以很好地处理变密度的不确定数据集的离群检测问题。而文献[6]在详细分析了高维数据对不确定数据离群点检测产生的影响后，提出了一种基于子空间划分和密度估计的离群点检测算法。在规模为 10 万、真实离群点占 5%的测试数据集上，分别运行两种算法。图 5 和图 6 比较了两种基于密度的离群检测算法的执行效率

和精确度。可以看出，RLOF 算法在处理低维数据时执行速度较快；但随着维度增加，基于 R2-tree 索引的查询成本增加，基于子空间划分的检测算法更占优势。RLOF 算法在检测精度上效果更好。

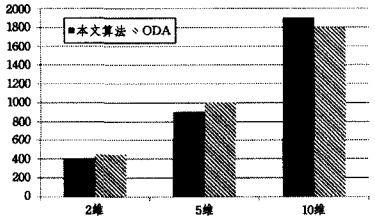

图 4　执行效率比较

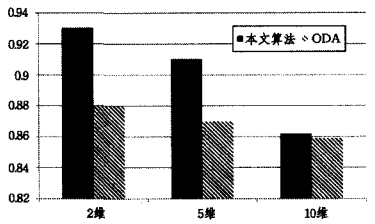

图 5　精确度比较

结束语　不确定数据的离群点检测是一个非常重要的研究领域，具有广泛的应用前景。本文提出了一种基于密度的局部离群点检测算法 RLOF。该方法通过引入 R2-tree 这种数据结构，在最小充分邻域的确定和数据集动态更新环节上缩短了计算时间，提高了整体算法的执行效率。实验表明，该算法一定程度上能够适应空间和海量数据环境下的离群点检测。下一步，将充分考虑不确定数据的属性级不确定性对算法的影响和面向不确定数据流的离群点检测算法。

参 考 文 献

[1] Breunig M M, Kriegel H P, Ng R T, et al. LOF: identifying density-based local outliers[J]. ACM Sigmod Record, 2000, 29(2): 93-104

[2] Tu L, Cui P, Tang K. A Density Grid-Based Clustering Algorithm for Uncertain Data Streams[C]// 2013 10th Web Information System and Application Conference (WISA). IEEE, 2013: 347-350

[3] Chawla S, Gionis A. k-means: A Unified Approach to Clustering and Outlier Detection[C]// SDM. 2013: 189-197

[4] Duforet-Frebourg N, Blum M G B. Bayesian Matrix Factorization for Outlier Detection: An Application in Population Genetics[M]// The Contribution of Young Researchers to Bayesian Statistics. Springer International Publishing, 2014: 143-147

[5] Cao K, Han D, Wang G, et al. An Algorithm for Outlier Detection on Uncertain Data Stream[M]// Web Technologies and Applications. Springer Berlin Heidelberg, 2013: 449-460

[6] Aggarwal C C, Philip S Y. Outlier Detection with Uncertain Data[C]// SDM. 2008: 483-493

[7] Yang C, Lin K I. An index structure for efficient reverse nearest neighbor queries[C]// 17th International Conference on Data Engineering, 2001. IEEE, 2001: 485-492

[8] Cao K, Han D, Wang G, et al. An Algorithm for Outlier Detection on Uncertain Data Stream[M]// Web Technologies and Applications. Springer Berlin Heidelberg, 2013: 449-460

[9] Hjaltason G R, Samet H. Distance browsing in spatial databases [J]. ACM Transactions on Database Systems (TODS), 1999, 24 (2): 265-318

[10] Aggarwal C C. On density based transforms for uncertain data mining[C]// IEEE 23rd International Conference on Data Engineering, 2007(ICDE 2007). IEEE, 2007: 866-875

[11] HU Cai-ping, QIN Xiao-lin. A Density-Based Local Outlier Detecting Algorithm[J]. Journal of Computer Research and Development, 2010(12): 2110-2116

[12] Zhou A Y, Jin C Q, Wang G R, et al. A survey on the management of uncertain data[J]. Chinese Journal of Computers, 2009, 32(1): 1-16

[13] Yu Hao, Wang Bin, Xiao Gang, et al. Distance-Based Outlier Detection on Uncertain Data[J]. Journal of Computer Research and Development, 2010, 47(3): 474-484

[14] Guttman A. R-trees: A dynamic index structure for spatial searching[M]. ACM, 1984

[15] Yu Min-min, Cheng Ning-jiang. Algorithm of Improved Top-k Query on Uncertain Data for Requirement Extension[J]. Computer Science, 2012, 39(6): 151-154

（上接第 159 页）

[7] 印鉴，王智圣，李琪，等. 基于大规模隐式反馈的个性化推荐[J]. 软件学报，2014，25(9)：1953-1966

[8] German U, Joanis E, Larkin S. Tightly Packed Tries: How to Fit Large Models into Memory, and Make them Load Fast, Too[C]// Proc. of the NAACLHLT Workshop. 2009, 8: 31-39

[9] Leskovec J, Lang K J, Mahoney M. Empirical comparision of algorithms for network community detection [C]// Proceedings of the 19th International Conference on World Wide Web. New York: ACM, 2010, 19: 630-655

[10] 徐志明，李栋，刘挺，等. 微博用户的相似性度量及其应用[J]. 计算机学报，2014(1)：207-218

[11] 贾大文，曾承，彭智勇，等. 一种基于用户偏好自动分类的社会媒体共享和推荐方法[J]. 计算机学报，2012(11)：2381-2391

[12] 张丰，王箭，赵燕飞，等. 社交网络中一种基于社区推荐的信任模型[J]. 计算机科学，2014，41(5)：168-172

[13] 王玙，高琳. 基于社交圈的在线社交网络朋友推荐算法[J]. 计算机学报，2014(4)：801-808

[14] Alan M, Massimiliano M, Krishnap G, et al. Bhattacharjee Bobby, Measurement and analysis of online social network[C]// Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement. SanDiego, CA, USA, 2007, 5: 29-45

[15] Chen Ke-han, Han Pan-pan, Wu Jian. User Clustering Based Social Network Recommentdation[J]. College of Computer Science and Technology, 2013, 3: 100-107

[16] Herlocker J. Evaluating collaborative filtering recom mender systems[J]. ACM Transactions on Information systens, 2004, 22 (1): 5-53

[17] 文俊浩，舒珊. 一种改进相似性度量的协同过滤推荐算法[J]. 计算机科学，2014，41(5)：68-71

[18] 高明，金澈清，钱卫宁，等. 面向微博系统的实时个性化推荐[J]. 计算机学报，2014(4)：963-975

[19] 海本斋，解瑞云. 基于贝叶斯网络的上下文推荐算法[J]. 计算机科学，2014，41(7)：275-278

[20] 朱宝，徐玲玉. 一种个性化推荐方法[J]. 计算机科学，2014，41 (11A)：294-297