基于Squeezer算法的数据流离群数据挖掘算法

王 超,倪志伟,朱小虎
(合肥工业大学,安徽 合肥 230009)

摘 要:由于数据流数据的动态性、时序性和数据量大等特点使得数据流上的数据挖掘变得更加困难和富有挑战。通过对Squeezer聚类算法的研究分析,并基于此算法提出了一种新的基于聚类的数据流离群数据检测算法O-Squeezer。把数据流看成一个随时间变化的过程,并将其分成许多数据分区,在每个数据块内用改进的O-Squeezer算法挖掘离群数据。理论分析和实验表明,算法可以有效发现数据流中的局部离群数据,算法是可行的。

关键词:数据挖掘;数据流离群数据;质心;Squeezer聚类算法
中图分类号:TP301.6 **文献标识码**:A **文章编号**:1673-629X(2008)01-0087-03

A Data Stream Outliers Detection Algorithm Based on Squeezer Cluster Algorithm

WANG Chao, NI Zhi-wei, ZHU Xiao-hu
(Hefei University of Technology, Hefei 230009, China)

Abstract: It is difficult to mine and analyse data streams, because data streams are dynamic, time sequence, have large amount of data. Based on Squeezer cluster algorithm, proposes a new algorithm O-Squeezer to mine outliers. Data streams will be divided into a lot of data sets, and in each data set the O-Squeezer algorithms will be used to detect outliers. Theoretic analysis and experimental results indicate that the algorithm is effective.

Key words: data mining; data stream outliers; centroid; Squeezer cluster algorithm

0 引 言

离群数据挖掘是数据挖掘领域的一个重要研究方向,主要是挖掘那些与数据集中大部分数据的一般特性不一致的数据[1]。在很多情况下,离群数据所蕴含的知识可能比常规数据所蕴含的知识更有价值。离群数据的检测方法主要有基于距离的方法、基于密度的方法、基于偏离的方法,但传统的离群数据挖掘算法应用在数据流这种新型的数据形式上的表现普遍不好,这主要是因为传统的离群数据挖掘算法大多是基于内存的,而数据流是连续到达的、近乎无限的有序数据序列,想要完整地或部分地保存过去的数据都是不可行的。

这些数据在时间维上是单调有序的,读取次数是受到严格限制的,只能被读取一次或者有限的几次。由于数据流的动态性和近乎无限的数据量,怎样在有限的存储空间中快速地处理这些数据并获得有价值的信息,成为了数据挖掘研究领域的一个重要课题。如何在大量的数据流数据中发现那些与众不同的,与其他数据有明显区别的数据,则显得尤为重要,因为在网络入侵检测、气象预报、电信和信用卡欺诈等众多应用领域有着广泛的需求。

Squeezer算法[2]不需要先验知识,基于数据之间的最大相似度或者最小距离,只需扫描数据集一遍即可将数据集分割成几个不同超球体,对数据集进行聚类。此算法也可以用于离群数据的挖掘,利用新数据与类中心数据的距离,以及一个预定的距离阈值来判断数据是否属于该聚类,从而判断新数据是否为离群数据。但是算法没有考虑聚类的局部密度,并且距离阈值不能够实时地动态更新,不能有效地发现聚类的局部离群数据。文中提出的算法在此基础上做了一些改进以适应数据流数据的特点,考虑了聚类的密度在离群数据检测中的作用,在数据流的每个数据分区内部使用改进的Squeezer算法,提高了离群数据挖掘的准确性。

收稿日期:2007-03-31
基金项目:国家自然科学基金项目(70631003);安徽省教育厅科研项目(2006sk010)
作者简介:王 超(1983-),男,安徽蚌埠人,硕士研究生,从事人工智能研究;倪志伟,教授,博士生导师,研究方向为管理信息化、决策科学与技术、软件工程。

1 相关工作

1.1 数据流模型

数据流由一系列按序到达的数据组成，也可看作是信息传输过程中经编码处理的数字信号串[3]。令 t 表示任一时间戳，α_i 表示在该时间戳到达的数据，数据流可以表示成 $\{\cdots,\alpha_{i-1},\alpha_i,\alpha_{i+1},\cdots\}$，对应描述了一个隐函数 $A:[1,\cdots,N]\rightarrow R^2$。

根据 α_i 对函数 A 描述的不同，数据流模型可分为三类：时间序列、Cash Register、Turnstile。

(1) 时间序列数据流，用来描述时间序列数据。如每分钟纳斯达克(NASDAQ) 成交量、每五分钟所观测到的 IP 流量。此时：

$$\alpha_i = (j, I_i) \tag{1}$$

其中 i 对应着递增的时刻。

(2)Cash Register 数据流，这是一类应用普遍的模型，类似收银机记录。如对 IP 地址的监控，同一 IP 资源可对一地分时传送或向多个地址传送。此时：

$$\alpha_i = (j, I_i), I_i \geq 0, A_i[j] = A_{i-1}[j] + I_i \tag{2}$$

(3)Turnstile 数据流，是由拥挤的地铁站中记录乘客出入的十字转门的启发而得到的。可有效研究动态的删除与插入操作，但很难得到有意义的界。此时：

$$\alpha_i = (j, U_i), I_i \geq 0, A_i[j] = A_{i-1}[j] + U_i \tag{3}$$

其中 U_i 可视作删除与插入符，数值可正可负。

上述 3 种流模型，前 2 种具有很好的实际意义，特别是时序数据流。Turnstile 模型具有较好的理论价值。

1.2 数据流离群数据检测

不同的算法对于离群数据的定义有所不同，但是目前广泛被接受的定义是 Hawkins 给出的形式化定义[4]。文中的算法也是基于此定义的。

定义 1 如果一个数据样本与其他样本之间存在足以引起怀疑的差异，则称其为离群数据。

与传统的静态数据不同，数据流数据具有数据量几乎无穷，且随着时间动态增加等特点。这些都要求在数据流离群数据挖掘中，关键是解决两个核心问题：一是如何在有限的内存空间中处理连续的、大量的随着时间不断地增长的数据流数据，因此必须提高算法的处理效率，改进算法的复杂度；二是由于数据量是“无限”的，因此不可能将所有的数据都存储起来，然后再对数据进行多次扫描。这就要求算法能够只扫描数据一次或有限的几次就能够发现离群数据。为了解决以上两个问题，必须在算法的精确性和高效性上做出平衡。

1.3 Squeezer 算法

Squeezer 算法是一种应用于大规模数据集的一种聚类算法，但它也可以用于数据流问题。此算法的基本思想是：①初始时，聚类集合为空，读入一个新数据；②以这个数据对象构造一个新的类；③若已到数据库末尾，则转⑥，否则读入新的数据对象，利用给定的距离定义计算它与每个已有类间的距离，并选择最小的距离；④若最小距离满足给定的阈值 r，则转②；⑤否则将该对象并入最小距离的类中，转③；⑥结束[5]。此算法本身也具有一定的离群数据处理能力。在聚类结束时，那些规模较小的类，在满足一定的条件的情况下就可以被当作离群数据来处理。但是此算法直接应用在数据流数据上时，在离群数据发现的精度和效率上并不理想。

在观察 Squeezer 算法对新数据的处理中发现：它利用新数据与类中心数据的距离，以及一个预定的距离阈值来判断数据是否属于该子聚类，但距离某一聚类中心近的点不一定就属于该聚类，也可能是一个离群数据。

如图 1 所示：点 A 在 Squeezer 算法下是满足聚类条件的，是一个正常的数据。但如图所示应该将其作为一个离群数据来处理。而点 B 在 Squeezer 算法下不满足聚类条件，被划作一个新类或被作为离群数据来处理，而事实上点 B 是应该划归此类的。造成这种情况的原因主要是因为：事先设定的阈值是固定的，无法对需要动态更改阈值的情况进行适应，所以才会出现这样的情况。

文中所提出的 O－Squeezer 算法是针对上述问题的改进，并且对数据流中的数据进行分区，每个分区由 N 个依次到达的数据流数据组成，使其成为适应数据流数据的一种基于聚类的离群数据挖掘算法。

图 1 不同密度情况下的离群点

2 O－Squeezer 算法设计

2.1 算法描述

该方法的优点是不需要关于样本数据集的知识。基于聚类算法的离群数据挖掘的基本思想是首先对样本数据进行聚类操作，然后检测无法归类的孤立点，这些孤立点就是离群数据[6－8]。

基于聚类算法的离群数据挖掘可分解成三个问题：

1)用聚类算法进行聚类操作,确定类别特征;

2)离群数据的确定:根据离群数据的定义并结合具体的对象,确定离群数据的度量标准;

3)检测离群数据。

O－Squeezer 算法正是基于以上三步骤设计的,主要有以下几个重要的参数和变量:固定阈值 T,新数据点与类内其它点的平均距离 d_{new},当前类质心[9]的平均距离 d_{avg},动态阈值 δ 和百分比 P。

算法描述如下:

对每一个新来的数据点 X_i,进行下面的处理:

在大小相等的数据分区 W_i 中根据预先设定的阈值 T 通过 O－Squeezer 聚类算法进行预运算,将 X_i 归入满足阈值 T 的条件聚类,并且获取该数据点与当前聚类中所有数据点的平均距离 d_{new},并将其与当前聚类的平均距离 d_{avg} 做比较。如果 d_{new} 小于 d_{avg} 与预先设定的百分比 P 的乘积,那么修改该聚类的聚类质心,将该数据点归入该聚类。如果 d_{new} 大于 d_{avg} 与预先设定的百分比 P 的乘积,那么计算该数据点周围的数据密度,并计算动态阈值 δ,再比较 d_{new} 与 δ 的大小以确定其是否为离群数据点。

否则,如果都不符合阈值 T 的条件的话,可以以此数据点为质心创建一个新的聚类。

在分区中的数据处理完时,将一些单个的质心点和较小规模的聚类都当作离群数据处理,并计算当前分区质心,用以判断数据流是否发生了概念转移。保存所有剩下的质心到下一个分区 W_{i+1}。

2.2　O－Squeezer 算法

```
O－Squeezer( T,P)
{
  在初始分区 W1 中,读入一个新的数据对象 X1 作为一个初始质心 Z1;
  FOR(数据流未尾){
    FOR(分区数据未结束){
      FOR(所有质心 Zi){
        通过距离定义,将依次到达的数据 Xj 与 Zi 比较,得到它们与 Zi 的距离 Dzij;
        IF(最小的距离 MIN(Dz1i) < T {
          数据点 Xj 属于 Zi 所在的类 Ci;
          计算 davg,dnew;
          IF(P * davg > dnew){
            Xj 取代 Zi 成为类 Ci 的新质心;
          }
          ELSE IF(P * davg < dnew){
        计算以 Dzij/2 为半径的超球体的数据密度 DENj;
        动态阈值 δj = K * davg * DENj;(K 为常数)
        IF(davg > δj){
          Xj 为离群数据点;
        }}}
        ELSE{
        Xj 作为一个新的类的质心 Zi;
        }}}
      单个质心数据点和较小规模的类作为离群数据处理;
      保存剩下的质心到下一数据分区 Wi+1;
      计算分区 Wi 的质心 WZj;
      比较 WZj 与 WZj-1,判断数据流是否发生概念转移;
      NEXT 分区;
  }}
```

3　实验分析

本节通过二维和四维模拟数据流测试数据集对 O－Squeezer 算法的效率和有效性进行测试和分析。二维模拟数据流分区数据点个数 N 为 600,四维模拟数据流分区数据点个数 N 为 500,T 取 0.4,P 取 10%,实验结果如图 2 所示。

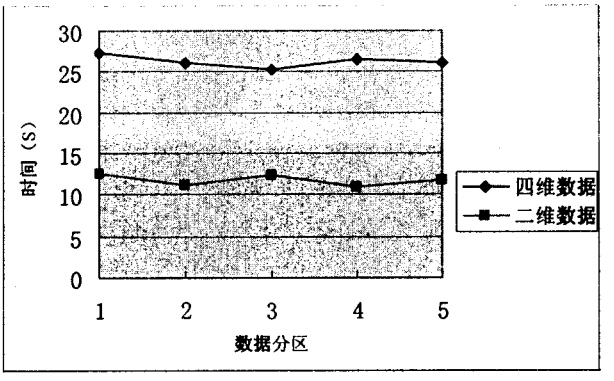

图 2　算法的运行效率

图中,在二维和四维模拟数据流中的第一个分区的运算时间普遍比其他分区多,这主要是因为其他分区可以直接利用初始分区的聚类质心进行离群数据探测,简化了算法的运行步骤。尽管在二维数据的每个分区中数据量比四维数据分区的要大,但算法在四维数据上的运行效率仍然要比在二维数据上低,这主要是因为高维数据所需要的运算时间比低维数据更多。通过表 1 可以看到 O－Squeezer 算法是可以准确地发现离群数据的。由实验结果可知,算法的效果和效率受参数 P,T 的取值的影响较大,这主要是因为文中的算法是基于 Squeezer 聚类算法的,而 Squeezer 聚类算法对于阈值 T 的取值是比较敏感的。

表 1　不同分区发现的离群数据个数

	分区 1	分区 2	分区 3	分区 4	分区 5
二维数据	9	9	12	8	11
四维数据	6	8	5	11	9

(下转第 92 页)

(a)噪声密度为 0.2 的 lena 图像

(b) 中值滤波输出图像

(c) 极值滤波算法输出图像

(d) 文中算法输出图像

图 3 lena 图像滤波结果比较

表 1 不同滤波算法对不同密度噪声的 PSNR

噪声密度	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
文中算法	34.33	32.50	29.63	26.88	23.34	20.30	17.41	15.01	11.56
中值滤波	29.01	26.66	21.86	17.78	14.48	11.57	9.28	7.41	6.01
极值滤波	34.87	30.59	24.08	19.67	15.95	12.74	10.17	8.03	6.31

从表中可以看出,在所有情况下,文中算法的滤波效果都要明显好于中值滤波算法。而同极值滤波算法比较,除了在噪声密度为 0.1 时文中算法的信噪比接近于极值滤波算法外,在其余情况,文中算法的滤波效果都要好于极值滤波算法。

参考文献:

[1] 黄煦涛. 二维数字信号处理Ⅱ - 变换与中值滤波器[M]. 北京:科学出版社,1985.

[2] Gonzalez R C, Woods R E. Digital Image Processing(影印版)[M]. 2nd ed. 北京:电子工业出版社,2002.

[3] Browningg D. The weighted median filter[J]. Commun Assoc Computer, 1984, 27(8): 807 - 818.

[4] Kos J, Neuvo Y. Center weighted median filters and their applications to image enhancement[J]. IEEE Trans Circuits Syst, 1991, 15: 984 - 993.

[5] Sun T, Neuro Y. Detai - preserving median based filters in image processing[J]. Pattern Recognition Letters, 1994, 15: 341 - 347.

[6] Wang Zhou, Zhang D. Progressive switching median filter for the remmval of impulse noise form highly corrupted images [J]. IEEE Trans Circuits and Systems Ⅱ: analog and digital signal processing, 1999, 46(1): 78 - 80.

[7] 邢藏菊,王守觉,邓浩江,等. 一种基于极值中值的新型滤波算法[J]. 中国图象图形学报,2001,6(6A):533 - 536.

[8] 秦 鹏,丁润涛. 一种基于排序阈值的开关中值滤波方法[J]. 中国图象图形学报,2004,9(4):412 - 416.

(上接第 89 页)

4 结论与未来工作

讨论了一种新的基于聚类的数据流离群数据挖掘方法——O - Squeezer 聚类算法,它可以较好地发现局部离群数据,并可以对局部离群阈值进行实时更改。试验结果显示,O - Squeezer 算法能够准确地发现离群数据。但是对于如何消除算法的效果和效率受参数 P, T 的取值的影响,将是下一步工作的方向。

参考文献:

[1] Han J, Kamber M. Data Mining[M]. New York: Morgan Kaufmann, 2001.

[2] HE Zengyou, XU Xiaofei. Squeezer: An Efficient Algorithm for Clustering Categorical Data[J]. Comput. Sci. & Technol, 2002, 17(5): 611 - 624.

[3] Gibbons P B, Matias Y. New sampling - based summary statistics for improving approximate query answers[C]//Proc of the ACM SIGMOD Int'l Conf on Management of Data. Seattle: ACM Press, 1998: 331 - 342.

[4] Hawkins D. Identification of Outliers[M]. London: Chapman and Hall, 1980.

[5] 蒋盛益,李庆华,李 新. 数据流挖掘算法研究综述[J]. 计算机工程与设计,2005,26(5):1130 - 1132.

[6] O'Callaghan L, Mishra N, Meyerson A. Streaming - data algorithms for high - quality clustering[C]//Proc of IEEE International Conference on Data Engineering. Washington, DC: IEEE Computer Society, 2002.

[7] Guha S, Mishra N, Motwani R. Clustering data streams[C]// Proc of IEEE Symposium on Foundations of Computer Science (FOCS'00). Redondo Beach: IEEE Computer Society, 2000: 71 - 80.

[8] Guha S, Meyerson A, Mishra N. Clustering data streams: Theory and practice[J]. Knowledge and Data Engineering, IEEE Transactions, 2003, 15(3): 515 - 528.

[9] Portnoy L, Esking L, Stolfo S J. Intrusion detection with unlabeled data using clustering[C]//Proc of ACM CSS Workshop on Data mining Applied to Security(DMSA - 2001). Philadelphia: ACM Press, 2001.