基于模糊聚类的数据流概念漂移检测算法

陈小东[1] 孙力娟[1,2] 韩 崇[1] 郭 剑[1,2]
(南京邮电大学计算机学院 南京 210003)[1]
(南京邮电大学江苏省无线传感网高技术研究重点实验室 南京 210003)[2]

摘 要 针对数据流中可能出现的概念漂移现象,采用改进的FCM算法进行模糊聚类,提出在大小可变的滑动窗口中通过度量相邻窗口之间的差异性来判断是否发生了概念漂移,并给出了相应的处理方法。实验表明该算法能够有效地检测出数据流中的概念漂移现象,具有很好的聚类效果和很高的时间效率。

关键词 概念漂移,数据流,模糊聚类,可变滑动窗口
中图法分类号 TP301 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.4.045

Detecting Concept Drift of Data Stream Based on Fuzzy Clustering

CHEN Xiao-dong[1] SUN Li-juan[1,2] HAN Chong[1] GUO Jian[1,2]
(College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)[1]
(Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)[2]

Abstract The phenomena of concept drift may occur in data stream, and how to detect it is very important in many applications. We used the improved version of FCM algorithm to cluster data in variable siding window, and measured the difference between adjacent windows to determine whether concept drift occurs. The result shows that our algorithm can detect concept drift in data stream effectively, and has great performance in clustering quality and time.

Keywords Concept drift, Data stream, Fuzzy clustering, Variable sliding window

1 引言

随着计算机、通信技术的发展以及大数据时代的到来,在过去的十几年中产生了许多无边界、连续、高速达到的数据对象,这种类型的数据一般被人们称为数据流。常见的数据流应用领域包括:网络入侵检测系统、传感器监测系统、股票交易、电子商务交易记录等[1]。由于数据量大、随时间连续到达等特点使得数据流很难完全放入内存进行处理,因此一般处理数据流的算法都需要满足以下两点要求[2]:

1)实时性,在后继数据到达之前完成前驱数据的挖掘任务;

2)单次遍历性,除了保存部分概要信息外,前驱数据在处理完后被丢弃,在后继数据的处理过程中无法再次被访问。

而在数据流的挖掘任务中不仅仅存在上述两个难题,很多流数据的模型概念常常会随着时间而发生变化,我们称之为进化的数据流。例如,消费者的购买习惯可能随着季节、可替代品的易获得性、商场促销等因素发生变化;通过路由器的IP地址分布可能随着一次潜在的网络入侵而发生改变等等。那些考虑概念漂移的挖掘算法能够使所得模型更加贴近实际的数据对象。如何有效地检测出数据流演化过程中的概念漂移现象对于及时发现异常现象、预测数据演化趋势都具有重要的意义。

聚类分析是一种常见的数据挖掘方法,它将给定的一组数据对象划分到不同的簇中,依据事先指定的相似性度量方法,使得簇内对象相似度高而簇间对象相似度低。处理数据流的聚类算法需要在满足时间和内存的限制下,连续、增量地聚类不断到达的数据点,并且能够快速检测出数据流的变化情况,发现何时旧簇消失、何时新簇形成。

本文在研究了一系列数据流上的聚类算法和概念漂移检测算法的基础之上,提出了一种使用模糊聚类检测数据流中的概念漂移现象的算法。本文第2节介绍现有的一些数据流聚类算法和概念漂移检测算法;第3节对数据流模糊聚类中所用到的数学符号进行定义;第4节给出本文算法的核心思想框架;第5节是本文算法的实验部分;最后总结全文。

2 相关工作

数据流中的概念漂移现象是指在不同的时间段中产生的数据点服从不同的分布模型。目前检测数据流中概念漂移现

到稿日期:2015-04-08 返修日期:2015-06-15 本文受国家自然科学基金(61171053,61300239),教育部博士点基金(20113223110002),中国博士后科学基金(2014M551635),江苏省博士后科研资助计划项目(1302085B),南京邮电大学引进人才科研启动基金(NY214013)资助。
陈小东(1992—),男,硕士生,主要研究方向为数据挖掘,E-mail:823265633@qq.com(通信作者);孙力娟(1966—),女,博士,教授,博士生导师,主要研究方向为演化计算、无线传感器网络和数据处理等;韩 崇(1985—),男,博士,讲师,主要研究方向为多媒体信息处理、数据挖掘;郭 剑(1978—),博士,副教授,硕士生导师,主要研究方向为无线传感器网络、无线多媒体传感器网络。

象所使用的主要手段是依据数据挖掘任务中的分类算法，依靠对未知数据预测错误率的大小判断数据流是否发生了概念漂移，如文献[3-5]。但是分类算法需要预先获得一些带有类标号的对象作为训练数据集，要想在原始的、不断进化的数据流中得到类标号是很困难的。本文采用聚类的方法检测数据流的漂移现象，放宽了算法的适用条件，使其能够适应更多的应用场景。

CluStream[6]是最早被提出来的应用于数据流的聚类算法。它将聚类过程分成两个部分：在线部分通过设定好的删除与合并原则增量更新微簇的特征向量；离线部分使用 K-means[7]算法将微簇聚类成最终簇，通过倾斜时间窗口使用户可以在不同时间粒度内发现不同的数据簇。

Phliip 等[8]提出的 Stream-Detect 算法使用在线聚类、离线分类的方法检测进化数据流的变化情况。在线部分比较新旧簇的聚类特征（包括聚类中心的均值、标准差、簇的平均大小和最大尺寸、最小簇中心等）、测量两者之间的偏离值，如果超出预先指定的阈值，则表示数据流有新的变化。通过离线分类处理，将数据流的变化类型分配到相应的事件或现象中。

文献[9]提出了一个可以聚类随时间变化的、离散的数据类型的框架。依据滑动窗口技术，通过统计前后窗口之间孤立点的个数以及不同簇的变化率，提出一种概念漂移检测算法，并能够可视化地显示出不同聚类结果之间的演化关系。但算法中所使用的滑动窗口大小固定，无法适应数据流的灵活变化情况。

李培培在基于聚类概念簇差异的概念漂移检测机制算法[10]中，采用每隔一个数据块调用一次聚类算法，通过比较前后两个聚类簇集合的平均距离与聚类簇半径之间的关系来判断是否发生了概念漂移。该算法与 Stream-Detect 的缺点一样明显，即不管数据流是否发生了漂移，每次都需要重新聚类，时间效率很低。曹付元[11]提出的时序分类数据的聚类算法就很好地解决了这个问题，该算法并不简单地对每个窗口进行聚类，而是对比前后窗口的数据对象，如果差异性大，则需要重新聚类，否则直接通过数据标签技术将数据划分到前一窗口最相似的簇中。文献[12]提出了一种检测增量数据集上的概念漂移算法。该算法对原始样本进行初始聚类，每当有一个新的数据点到达时，则判断其是否发生漂移。

上述文献在检测数据流的概念漂移情况时所使用的聚类算法都属于硬聚类算法，即一个数据点对象不是完全属于这个簇，就是完全属于另一个簇。这种算法是不具有一般意义的，因为在模糊逻辑中，一个数据对象可能同时属于多个簇。

FCM[13]算法是一个完整数据集上的模糊聚类算法。该算法通过不断迭代计算簇中心和隶属度矩阵，来求解聚类误差平方和 SSE 的最小值。文献[14]提出的 SWFCM 算法是对 FCM 算法的扩展，使之应用于数据流之上。依据数据流的到达顺序将其划分成连续的数据块，每一部分使用 FCM 算法处理，将所得的簇中心点和该部分所有数据点在该簇的隶属度之和（作为权值）传递到下一部分，依次类推，最终得到数据流上的簇中心点。文献[15]对 SWFCM 算法进行了扩展，通过对后继数据增大权值的方法来适应数据流的漂移情况。但该算法并没有考虑概念漂移的实际发生情况，只是一种减小历史数据影响的自适应调整策略，并不能达到检测概念漂移的目的。

本文提出的基于模糊聚类的数据流概念漂移检测算法，使用改进 FCM 算法的初始中心点的选择方式来加速聚类过程、提高聚类质量，同时依据变化的滑动窗口提供一种检测动态数据流中是否发生概念漂移的方法以及检测之后相应聚类簇的更新方式。

3 符号定义

本节将介绍算法中使用到的一些数学符号和模型定义。

定义 1（数据流模型） 数据流是由一串连续到达的数据点组成，形如 $s_1, s_2, \cdots, s_n, \cdots$，其中每个数据点 s_j 由 p 个属性组成，即 $s_j^1, s_j^2, \cdots, s_j^p$，表示该数据点的属性值。

定义 2（模糊簇） 模糊簇 C_i 是数据集 $s_1, s_2, \cdots, s_n$ 的一个子集，其中每个数据点 s_j 都有一个属于 C_i 的介于 0 到 1 之间的隶属度[16]。

定义 3（隶属度矩阵） 隶属度 U 是一个 $m \times n$ 的矩阵，其每行表示一个模糊簇，每列表示一个数据点。u_{ij} 表示第 j 个数据点在第 i 个模糊簇中的重要程度[16]。

性质 1 对于任意的数据点 s_j 及模糊簇 C_i，都有 $0 \leqslant u_{ij} \leqslant 1$。

性质 2 对于任意的数据点 s_j，都有 $\sum_{i=1}^{m} u_{ij} = 1$。

定义 4（模糊簇中心） 模糊簇 C_i 的簇中心与每个数据点在该簇中所占的比重有关[17]：

$$v_i = \frac{\sum_{j=1}^{n} (u_{ij}^b \times s_j)}{n} \tag{1}$$

其中，$b \geqslant 1$ 是 FCM 算法中的加权指数，簇中心 v_i 也是由 p 个数据属性组成。

定义 5（目标函数） 聚类的目标函数又称为误差平方和，是评价聚类结果对数据集拟合程度的重要标准之一。模糊聚类的误差平方和表达式如下：

$$J_{SSE} = \sum_{i=1}^{m} \sum_{j=1}^{n} u_{ij}^b \times dist(v_i, s_j) \tag{2}$$

其中，$dist(v_i, s_j)$ 表示模糊簇 C_i 的中心点与数据点 s_j 的差异性，在 FCM 算法中使用两者之间的欧氏距离来替代，即：

$$dist(v_i, s_j) = \sqrt{\sum_{k=1}^{p} (v_i^k - s_j^k)^2} \tag{3}$$

J_{SSE} 的值越小，表示数据集的拟合程度越好，聚类质量越高。为求得式(2)的最小值，结合性质 2，可得到隶属度 u_{ij} 的计算式：

$$u_{ij} = \frac{1}{\sum_{k=1}^{m} \left(\frac{dist(v_i, s_j)}{dist(v_k, s_j)} \right)^{\frac{2}{b-1}}} \tag{4}$$

定义 6（滑动窗口） 滑动窗口是用来处理数据流挖掘任务的常规方法，依据数据点到达的先后顺序，将其依次划分到大小不同的窗口中，记为 $B_1, B_2, \cdots, B_n, \cdots$，滑动窗口 B_i 中数据点的个数记为 N_i。

定义 7（概念漂移） 概念漂移是指相邻窗口 B_i 与 B_{i+1} 中的数据点差异性超过用户预先指定的阈值，即满足下式：

$$J'/N_{i+1} > \beta * J_{SSE}/N_i \tag{5}$$

其中，β 是用户指定的阈值参数，J_{SSE} 是窗口 B_i 的目标函数，J' 是窗口 B_i 中的簇中心和窗口 B_{i+1} 数据点之间的误差平方和。

4 基于模糊聚类的数据流概念漂移检测算法

本文提出的基于模糊聚类的数据流概念漂移检测算法首先使用改进的FCM算法对初始滑动窗口进行聚类，对比随后到达的数据点与前一阶段所形成簇的差异性，测量结果如果大于预先指定的阈值则判定出现了概念漂移，需要重新聚类当前窗口中的数据以形成新的簇，否则将当前窗口的数据点融合到上一窗口最相似的簇中，同时更新相关簇信息。在检测的过程中，动态改变滑动窗口的大小以适应数据流中概念漂移出现的频率。

4.1 改进的模糊C均值算法(REFCM)

FCM算法与K-means算法很相似，其主要思想如表1所列。

表1 FCM算法

1. 选定数据集聚类簇个数。
2. 随机分配簇中每个数据点的隶属度值，要求满足性质1和性质2。
3. 重复如下步骤直至算法收敛(即前后两次迭代误差平方和之差的绝对值不大于ε，其中ε为预先给定的灵敏度阈值)：
①依据式(1)计算每个簇的中心；
②对于每个数据点，使用式(4)计算隶属度矩阵。

FCM是一种局部优化搜索算法，若初始隶属度分配不恰当，容易陷入局部最优点。本文提出了一种新的改进的FCM算法(Reform Fuzzy Clustering Method，REFCM)。相比于FCM算法，改进的FCM算法的优点主要表现在对初始中心点选择的变化上，其思想如表2所列。

表2 REFCM算法

1. 随机选择一个数据点作为第一个簇的初始中心点。
2. 重复如下步骤，直至所有簇的初始中心点都选取完毕：
①计算剩余数据点与已经得到的簇中心之间的最短距离；
②将最短距离依据大小顺序排列，然后按轮盘的方式选出下一个簇的初始中心点，即距离越大，在轮盘上所占的面积越大，被选中的概率越大。
3. 对于选出的初始簇中心点，依据式(4)计算每个数据点的初始隶属度。
4. 余下过程按照FCM算法中步骤3处理。

相比于FCM算法中所有的初始簇中心都是随机分配的，REFCM算法中只是第一个初始中心是随机的，尽最大努力减小算法的不确定性。REFCM算法优先选择相互间距离较远的点作为初始簇中心，这符合簇内相似度尽可能高、簇间相似度尽可能低的聚类要求，能够加快搜索速度、提高效率。

4.2 可变大小的滑动窗口

在检测数据流前后窗口之间是否发生概念漂移的过程中，滑动窗口大小的设定是很重要的。正如文献[9]所述，在不同的应用场景中窗口大小的参数设定要求会有所不同：在不稳定的数据流应用中需要设定一个尺寸相对较小的滑动窗口，以捕获频繁的概念漂移现象；在相对稳定的数据流场景中最好设定大一点的窗口尺寸，用以加快数据流的处理过程。

这些都需要预先对数据的采集场景有一定的了解，无法应对一般性的概念漂移检测任务。更加严重的问题在于有些数据流可能在某一段时间内相对稳定，而在另一个时间段内变化得又特别频繁，若与文献[9]等算法一样将滑动窗口的尺寸设为某个固定值，无论或大或小，都无法有效、及时地检测出概念漂移现象。

本文采用动态变化的滑动窗口可以有效地解决这个问题：在聚类过程中如果发现了概念漂移，则减小滑动窗口；否则增大其尺寸以减小数据流的处理时间。具体做法如表3所列。

表3 滑动窗口调整策略

1. 定义最大窗口尺寸 N_{max} 和最小窗口尺寸 N_{min}，设当前窗口大小为 $N_{current}$。
2. 第一个和第二个滑动窗口的大小设定为 N_{min}。
3. 如果窗口1到窗口2未检测出概念漂移，则下一个窗口的大小设为 $\min\{N_{max}, 2*N_{current}\}$；否则，将下一个窗口尺寸重新定为 N_{min}。
4. 依次类推，定义其他的滑动窗口大小。

4.3 算法框架

本小节将详细介绍本文算法的核心框架，即如何检测变化的滑动窗口中是否出现了概念漂移现象，并提供检测后的相应处理方法。

假设已经定义好了最大窗口尺寸 N_{max} 和最小窗口尺寸 N_{min}，按照表3所述，窗口1中是数据流中的前 N_{min} 个数据点，采用改进的FCM算法对该初始窗口进行模糊聚类，在聚类结果中保存四元组：$\langle n, v, U, J\rangle$，其中 n 是窗口中数据点的个数，v 代表模糊簇中心，U 表示隶属度矩阵，J 是REFCM算法中最后趋于收敛的误差平方和。

依据表3计算下一个滑动窗口的大小，假设为 $N_{current}$，首先计算当前窗口中的数据点与上一个窗口形成的簇中心之间的误差平方和 J'，即通过式(4)计算新数据点与旧簇之间的隶属度矩阵 U'，再通过式(2)计算 J'。考虑到前后两个窗口的尺寸可能不一样，并不能直接比较 J 和 J' 之间的关系来判断是否发生了概念漂移，我们使用“单位误差平方和”来消除窗口尺寸不同的影响，即若 $J'/N_{current} > \beta * J/n$，则表明前一个窗口形成的簇相对后一个窗口中的数据点偏差较大，出现了漂移；否则就判断没有出现概念漂移。其中 β 是用户指定的大于1的阈值参数。

如果前后窗口出现了概念漂移，则对后一窗口重新使用REFCM算法聚类，并用新的四元组 $\langle n_{new}, v_{new}, U_{new}, J_{new}\rangle$ 替换前一阶段的，以反映数据的最新分布情况；如果检测完后发现没有出现概念漂移，则按照表4所提供的方法使用上一窗口的四元组和当前窗口中的新数据点共同更新簇信息。

表4 未发生概念漂移时四元组的更新方法

输入：上一窗口的四元组〈n，v，U，J〉和当前窗口的数据点个数 $N_{current}$、U'、J'
输出：更新后的四元组〈n_{new}，v_{new}，U_{new}，J_{new}〉
1. 用当前窗口的数据点个数替换 n_{new}，即 $n_{new}=N_{current}$。
2. 使用当前窗口所计算的 J' 与上一窗口中的 J 同时更新 J_{new}，即 $J_{new}=n_{new}*(J'+J)/(N_{current}+n)$。
3. 使用当前窗口的数据点和上一窗口的簇中心 v 及隶属度矩阵 U 同时更新 v_{new}，即 $v_{new,i}=\dfrac{\sum_{j=1}^{n} u_{ij}*v_i+\sum_{k=1}^{N_{current}} u'_{ik}*s_k}{\sum_{j=1}^{n} u_{ij}+\sum_{k=1}^{N_{current}} u'_{ik}}$
4. 使用当前窗口的 U' 替换 U_{new}，即 $U_{new}=U'$。

依据表4更新四元组 $\langle n_{new}, v_{new}, U_{new}, J_{new}\rangle$ 后，继续按照表3确定下一滑动窗口的尺寸，依次类推，直至数据流处理结束。图1给出了算法的整个流程。

Stream-Detect等算法对每个窗口都进行聚类从而处理比较所得簇之间的差异性，而本文所提算法仅仅对那些发生了概念漂移的窗口重新聚类，没有漂移的窗口只是简单地更新四元组，仅相当于FCM算法中的一次迭代，简化了算法的处理过程，提高了时间效率。

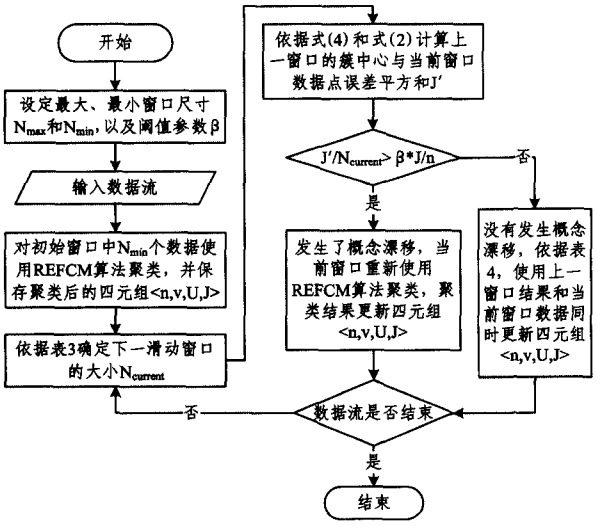


图 1　基于模糊聚类的数据流概念漂移检测算法流程

5　实验结果

本文使用 Matlab 软件实现了基于模糊聚类的数据流概念漂移检测算法，进行了以下 3 组实验：1)验证提出的算法能否有效地检测出变化滑动窗口中的数据流概念漂移现象；2)对比在相同的框架中使用 REFCM 算法与 K-means 算法在聚类效果和时间效率上的差异；3)对比每个窗口，先判断是否发生漂移，再确定是否需要重新聚类与每个窗口都直接重新聚类两种做法的聚类效果与时间效率上的差异。所有实验都是在一个 2GB 内存、酷睿 2 CPU、操作系统为 Windows 7 的主机上运行的。

5.1　算法有效性验证

要想验证算法是否能够有效地检测出数据流中的概念漂移现象，必须要有已知概念漂移在何处发生的数据集合。本文使用人工合成的 3 维数据点来模拟在特定窗口发生漂移的数据流，具体有以下 3 种类别的数据：

1)Class1：均值为[0 0 0]、协方差为[0.3 0 0；0 0.35 0；0 0 0.3]的高斯分布数据；

2)Class2：均值为[1.25 1.25 1.25]、协方差同上的高斯分布数据；

3)Class3：均值为[−1.25 1.25 −1.25]、协方差同上的高斯分布数据。

对 3 种类别的数据每种各取 100 个点，数据分布如图 2 所示。

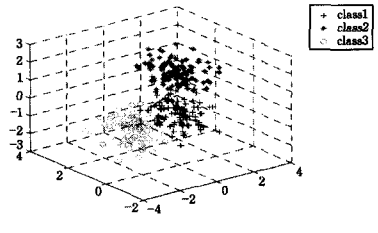


图 2　3 种不同类别数据的分布图

从图 2 可以看出 3 种类别的数据并不是完全分离的，概念模型的区分不是非常明显，这给检测增加了一定的难度。本文指定最大滑动窗口 $N_{max}=600$，最小滑动窗口 $N_{min}=100$，考虑到滑动窗口的自动调整策略，设计了如表 5 所列的合成数据流。

表 5　可变滑动窗口的数据流

类别＼窗口	1	2	3	4	5	6	7	8	9	10
Class1	50	30	100	280	0	0	0	200	30	140
Class2	50	70	100	120	500	50	60	0	0	0
Class3	0	0	0	0	100	50	140	200	70	60

参考表 3 的滑动窗口调整策略，对表 5 的设计进行如下说明：窗口 1 从最小尺寸 $N_{min}=100$ 开始，窗口 2 的尺寸也是 100，窗口 1 到 2 没有发生漂移，故窗口 3 尺寸扩大一倍为 200。同样，窗口 4 为 400，由于窗口 3 到 4 也没漂移，因此窗口 5 应该为窗口 4 的两倍，又因为最大窗口 $N_{max}=600$，所以窗口 5 尺寸为 600，而窗口 4 到窗口 5 发生了漂移，所以窗口 6 的尺寸又变为最小的 $N_{min}=100$，依次类推设计完成整个数据流。

阈值参数 β 的设定依赖于数据流自身的分布特性以及用户主观意愿，本文参照文献[9,10]，将前后窗口判断是否发生漂移的阈值参数 β 设为 1.3，簇个数设为 2，REFCM 算法的参数设置同 FCM 算法，通过在合成的数据流上进行实验，滑动窗口尺寸的变化如图 3 所示。

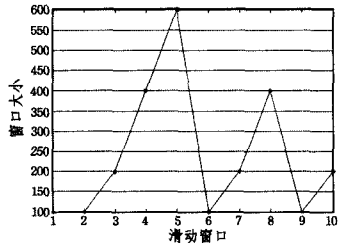


图 3　滑动窗口尺寸的变化

图 3 验证了本文算法的有效性，即当没有漂移发生时，窗口慢慢增长至最大尺寸；而在第 4 到第 5 个窗口检测出了概念漂移现象，所以下一个窗口尺寸又变为最小。算法准确地检测出了数据流中概念漂移发生的时刻(第 4 到第 5 个窗口和第 7 到第 8 个窗口)，符合预期设定。在检测过程中动态改变滑动窗口大小以灵活适应数据流中概念漂移发生频率的变化。

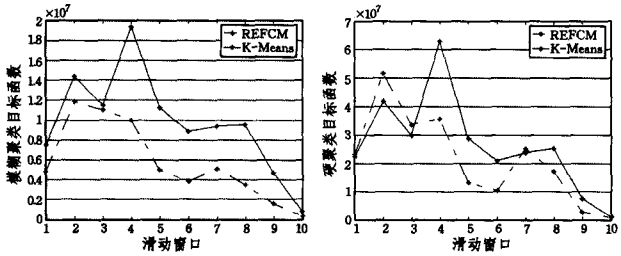
5.2　算法性能对

本小节将本文算法中使用到的 REFCM 算法替换成 K-means 算法来对比算法性能。考虑到 REFCM 算法是模糊聚类算法，而 K-means 是经典的硬聚类算法，本文将同时使用两种聚类效果评价函数：(1)模糊聚类下的误差平方和，即式(2)；(2)硬聚类下的误差平方和，即式(2)中的 $u_{ij}\in\{0,1\}$，将数据点全部分配到隶属度最大的簇中，而在其余簇该点的隶属度为 0。

本节采用的实验数据集是垃圾邮件数据集 Spambase[18]，该数据集有 4601 个数据点，每个数据点有 57 个维度。为防止因为概念漂移检测结果可能出现的不一致，导致 REFCM 和 K-means 算法对应窗口的数据集不一样，使得测量结果没有可比性，本小节暂且使用固定的滑动窗口处理，每个窗口大小设为 500，最后一个窗口大小为 101，共 10 个窗口，簇个数为 4，其余参数设置同 5.1 节，实验结果如图 4、图 5 所示。

针对 Spambase 数据而言，REFCM 与 K-means 算法检测到的概念漂移情况是一样的，都是在第 1 到第 2、第 2 到第 3、

第3到第4个窗口出现了漂移，其余窗口未发现。从图4、图5中可以看出，无论是在哪种误差平方和的评价标准下，REFCM算法的目标函数整体上都是小于K-means的，聚类效果更好，而且检测处理时间更短，本文提出的算法展现出了更加优越的性能。



(a)REFCM与K-Means的模糊聚类目标函数对比　(b)REFCM与K-Means的硬聚类目标函数对比

图4　REFCM与K-means算法误差平方和的对比

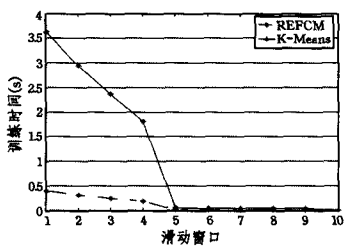
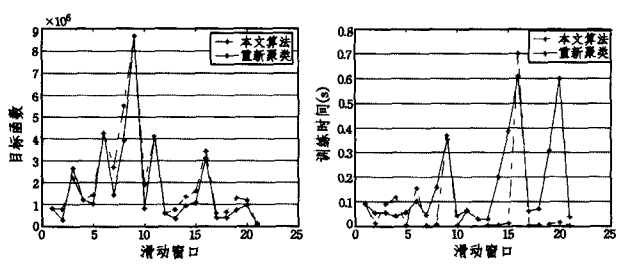


图5　REFCM与K-means算法聚类时间的对比

5.3　窗口重新聚类

在本文的算法框架中，除初始窗口外，每个窗口先判定是否发生了漂移，只对那些发生了漂移的窗口进行重新聚类，而没有发生漂移的窗口只是通过表4简单更新四元组$\langle n_{new}, v_{new}, U_{new}, J_{new}\rangle$。如果同文献[8-10]，无论数据流是否发生漂移，对每个到达的窗口都进行聚类，那么结果又将是如何呢？

本小节继续使用本文提出的算法框架处理Spambase数据集，可变的滑动窗口设置如5.1节所述，簇个数为4，阈值参数$\beta=1.5$，对得到的每个滑动窗口中的数据都重新采用REFCM聚类，对比本文算法与每个窗口都重新聚类两种方法，实验结果如图6所示。



(a)本文算法与都重新聚类的目标函数对比　(b)本文算法与都重新聚类的时间对比

图6　本文算法与每个窗口都重新聚类的性能对比

从图6可以看出，相对于文献[8-10]中提到的算法，即每个窗口都需要重新聚类来判断是否发生了概念漂移，本文提出的算法虽然在误差平方和方面稍稍逊色一点，但是对那些没有发生漂移的窗口的处理速度提高了很多，极大地提高了算法的时间效率。

结束语　本文提出了一种基于模糊聚类的数据流概念漂移检测算法，使用改进的FCM算法作为基准模糊聚类算法，解决了在可变滑动窗口中检测是否有概念漂移现象发生以及如何处理的问题。最后通过实验从概念漂移检测的有效性、模糊聚类效果及时间效率3个方面展现出了该算法的优越性。接下来的研究内容主要分为以下几类：如何处理数据流概念漂移检测中可能出现的噪声点问题；改进滑动窗口的调整策略，如可以通过定义潜在的概念漂移，使得滑动窗口缓慢减小等。

参考文献

[1] Chen Yi-xin, Li Tu. Density-Based Clustering for Real-Time Stream Data[C]//Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. California, 2007: 133-142

[2] Silva J A, Faria E R, Barros R C, et al. Data stream clustering: A survey[J]. ACM Computing Surveys, 2013, 46(1): 1-13, 31

[3] Dongre P B, Malik L G. A review on real time data stream classification and adapting to various concept drift scenarios[C]// 2014 IEEE International Advance Computing Conference (IACC). IEEE, Gurgaon, 2014: 533-537

[4] Padmalatha E, Reddy C R K, Rani B P. Classification of Concept Drift Data Streams[C]// 2014 International Conference on Information Science and Applications (ICISA). IEEE, Hainan, 2014: 1-5

[5] Wang H, Fan W, Yu P S, et al. Mining concept-drifting data streams using ensemble classifiers[C]//Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Washington, 2003: 226-235

[6] Aggarwal C C, Han J, Wang J, et al. A framework for clustering evolving data streams[C]//Proceedings of the 29th International Conference on Very Large Data Bases-volume 29. VLDB Endowment, 2003: 81-92

[7] Lloyd S. Least squares quantization in PCM[J]. IEEE Transactions on Information Theory, 1982, 28(2): 129-137

[8] Gaber M M, Yu P S. Detection and classification of changes in evolving data streams[J]. International Journal of Information Technology & Decision Making, 2006, 5(4): 659-670

[9] Chen H L, Chen M S, Lin S C. Catching the trend: A framework for clustering concept-drifting categorical data[J]. IEEE Transactions on Knowledge and Data Engineering, 2009, 21(5): 652-665

[10] Li Pei-pei. Concept Drifting Detection and Classification on Data Streams [D]. Hefei: Hefei University of Technology, 2012 (in Chinese)
李培培. 数据流中概念漂移检测与分类方法研究[D]. 合肥：合肥工业大学，2012

[11] Cao Fu-yuan. Studies on Clustering Algorithms for Categorical Data[D]. Taiyuan: Shanxi University, 2010 (in Chinese)
曹付元. 面向分类数据的聚类算法研究[D]. 太原：山西大学，2010

[12] Hu Wei. Research and realization of a web information extraction and knowledge presentation system [J]. Application of Computer System, 2013, 22(5): 116-121 (in Chinese)
胡伟. 一种改进的动态k-均值聚类算法[J]. 计算机系统应用，2013, 22(5): 116-121

(下转第251页)

值，随后误差增大。图 5(b)所示为 MovieLens1M 数据集近邻大小对 *MAE* 效果的影响。*Top-k* 的选取范围为 20～100，随着 *Top-k* 取值的增长，当 *Top-k* 的取值大于 40 后，*MAE* 的值没有呈现出明显的变化。通过分析可以证明，数量相当的近邻能够提供足够的信息，过多的近邻可能带来相对较多的不相关信息，从而使得精度下降。

结束语 本文旨在解决当前研究新项目和用户对其评分较少的项目的冷启动问题的局限。由于项目属性信息能够改善预测评分的准确性，本文将耦合相似度度量方法与矩阵分解技术相结合来优化推荐系统的预测能力。通过分析和实验，证明了耦合相似关系为寻找相似项目提供了较准确的信息。

在接下来的工作中，需要收集更多带有属性信息的相关数据集，以进一步改进算法。同时，由于本文没有探究冷启动用户的表现，因此考虑加入社交关系来丰富推荐框架以及更有效的算法都有待进一步研究。

参 考 文 献

[1] Balabanovic M, Shoham Y. Fab: Content-based collaborative filtering [J]. Communications of the ACM, 1997, 40(3): 66-72

[2] Breese J S, Heckerman D, Kadie C. Empirical analysis of predictive algorithms for collaborative filtering [C]// Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence. 1998: 43-52

[3] Cao L B, Ou Y, Yu P S. Coupled behavior analysis with applications [J]. IEEE Transactions on Knowledge and Data Engineering, 2012, 24(8): 1378-1392

[4] Gantner Z, Drumond L, Freudenthaler C, et al. Learning attribute-to-feature mappings for cold-start recommendations [C]// Proceedings of the 10th International Conference on Data Mining. 2010: 176-185

[5] Jaschke R, Marinho L, Hotho A, et al. Tag recommendations in folksonomies [C]// Proceedings of the 11th Conference on European Conference on Principles and Practice of Knowledge Discovery in Databases. 2007: 506-514

[6] Hotho A, Jaschke R, Schmitz C, et al. FolkRank: A Ranking Algorithm for Folksonomies [J]. LWA, 2006, 1: 111-114

[7] Koren Y. Collaborative filtering with temporal dynamics [J]. Communications of the ACM, 2010, 53(4): 89-97

[8] Lee D D, Seung H S. Algorithms for non-negative matrix factorization [C]// Proceedings of the 14th Conference on Advances in Neural Information Processing Systems. 2001: 556-562

[9] Middleton E, Shadbolt R, De Roure C. Ontological user profiling in recommender systems [J]. ACM Transactions on Information Systems, 2004, 22(1): 54-88

[10] Linden G, Smith B, York J. Amazon. com Recommendations: Item-to-item Collaborative Filtering [J]. IEEE Internet Computing, 2003, 7(1): 76-80

[11] Lotfi H, Fallahnejad R. Imprecise shannon's entropy and multi attribute decision making [J]. Entropy, 2010, 12(1): 53-62

[12] Levandoski J, Sarwat M, Eldawy A, et al. LARS: A Location-Aware Recommender System [C]// Proceedings of the 28th Conference on Data Engineering. 2012: 450-461

[13] McAuley J, Leskovec J. From amateurs to connoisseurs: Modeling the evolution of user expertise though online reviews [C]// Proceedings of the 22th International Conference on World Wide Web. 2013: 897-908

[14] Ma H, King I, Lyu M R. Learning to recommend with social trust ensemble [C]// Proceedings of the 32th Conference on Research and Development in Information Retrieval. 2009: 203-210

[15] Ma H, Zhou D, Liu C. Recommender system with social regularization [C]// Proceedings of the 4th Conference on Web Search and Data Mining. 2011: 287-296

[16] Nguyen J J, Zhu M. Content-boosted matrix factorization techniques for recommender systems [J]. Statistical Analysis and Data Mining, 2013, 6(4): 286-301

[17] Paterek A. Improving regularized singular value decomposition for collaborative filtering[C]// Proceedings of the 13th International Conference on Knowledge Discovery and Data Mining. 2007: 5-8

[18] Sarwar B, Karypis G, Riedl J. Item-based collaborative filtering recommendation algorithms [C]// Proceedings of the 10th International Conference on World Wide Web. 2001: 285-295

[19] Sarwar M, Karypis G, Konstan J, et al. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering [C]// Proceedings of the 5th International Conference on Computer and Information Technology. 2002

[20] Salakhutdinov R, Mnih A. Probabilistic matrix facotorization [C]// Proceedings of the 20th Conference on Neural Information Processing Systems Foundation. 2007: 1257-1264

[21] Salakhutdinov R, Mnih A. Bayesian probabilistic matrix factorization using markov chain monte carlo [C]// Proceedings of the 25th Conference on International Conference on Machine Learning. 2008: 880-887

[22] Wang J, De Vries A P, Reinders M J T. Unifying user-based and item-based collaborative ltering approaches by similarityfusion [C]// Proceedings of the 29th Conference on Research and Development in Information Retrieval. 2006: 501-508

[23] Song Y, Cao L B, Wu X, et al. Coupled behavior analysis for capturing coupling relationships in group-based market manipulations [C]// Proceedings of the 18th International Conference on Knowledge Discovery and Data Mining. 2012: 976-984

(上接第 223 页)

[13] Bezdek J C, Ehrlich R, Full W. FCM: The fuzzy c-means clustering algorithm[J]. Computers & Geosciences, 1984, 10(2): 191-203

[14] W Ren-xia, Y Xiao-ya, S Xiao-ke. A Weighted Fuzzy Clustering Algorithm for Data Stream[C]// International Colloquium on Computing, Communication, Control, and Management, 2008 (CCCM'08). 2008: 360-364

[15] Jaworski M, Duda P, Pietruczuk L. On fuzzy clustering of data streams with concept drift[C]// Artificial Intelligence and Soft Computing. Springer Berlin Heidelberg, 2012: 82-91

[16] Jiawei H, Micheline K, Jian P. Data Minging: Concepts and Techniques(Third Edition)[M]. San Francisco: Morgan Kaufmann Publishers, 2012: 323-350

[17] Shi Feng, Wang Hui, Yu Lei, et al. Matlab Intelligent Algorithm: Analysis of 30 Cases [M]. Beijing: Beihang University Press, 2011: 188-196(in Chinese)
史峰，王辉，郁磊，等. Matlab 智能算法：30 个案例分析[M]. 北京：北京航天航空大学出版社，2011：188-196

[18] David A. UCI Machine Learning Repository[OL]. http://archive. ics. uci. edu/ml/datasets. html