ISSN 1004-9037,CODEN SCYCE4
Journal of Data Acquisition and Processing Vol. 37,No. 4,Jul. 2022,pp. 894－908
DOI:10. 16337/j. 1004-9037. 2022. 04. 017

http://sjcj. nuaa. edu. cn
E-mail:sjcj@nuaa. edu. cn
Tel/Fax:＋86-025-84892742

数据流聚类算法研究

朱颖雯[1,2]，陈松灿[1]

(1. 南京航空航天大学计算机科学与技术学院，南京 211106；2. 三江学院计算机科学与工程学院，南京 210012)

摘　要：许多应用程序会产生大量的流数据，如网络流、web点击流、视频流、事件流和语义概念流。数据流挖掘已成为热点问题，其目标是从连续不断的流数据中提取隐藏的知识/模式。聚类作为数据流挖掘领域的一个重要问题，在近期被广泛研究。不同于传统的静态数据聚类问题，数据流聚类面临有限内存、一遍扫描、实时响应和概念漂移等许多约束。本文对数据流挖掘中的各种聚类算法进行了总结。首先介绍了数据流挖掘的约束；随后给出了数据流聚类的一般模型，并描述了其与传统数据聚类之间的关联；最后提出数据流聚类领域中进一步的研究热点和研究方向。

关键词：聚类；数据流；数据流聚类；流挖掘

中图分类号：TP391　　**文献标志码**：A

Research on Data Stream Clustering Algorithms

ZHU Yingwen[1,2], CHEN Songcan[1]

(1. College of Computer Science and Technology, Nanjing University of Aeronautics & Astronautics, Nanjing 211106, China; 2. College of Computer Science and Engineering, Sanjiang University, Nanjing 210012, China)

Abstract: Nowadays, developments of technology have allowed the generation of huge amounts of streaming data, such as network traffic flows, web click stream, video stream, event stream and semantic concept stream. Therefore, data stream mining has become a hot research topic and its goal is to extract hidden knowledge/patterns from continuous stream data. Clustering, as one of the most important problems in stream mining, has been highly explored recently. However, data stream clustering algorithms differ from traditional static data clustering algorithms in many aspects, and have more constraints such as bounded memory, single-pass, real-time response and concept-drift detection. In this paper, we survey the state-of-the-art data stream clustering algorithms. Firstly, mining constraints are identified. Then a general model for stream clustering is given, and its association with traditional data clustering is described. Finally, some further research issues in this domain are put forward.

Key words: clustering; data stream; data stream clustering; stream mining

引　　言

随着技术的发展，包括传感器在内的越来越多的设备正在成为互联设备，并不断地生成数据流[1-3]。例如：每天Google都要处理超过35亿的搜索；NASA卫星产生约4 TB的图片；沃尔玛超市每天产生超

基金项目：国家自然科学基金重点项目(61732006)。
收稿日期：2021-12-14；**修订日期**：2022-02-09

过2 000万笔交易。数据流不事先存放在存储介质中,具有快速、时序和海量等特征。数据流的特性使得传统数据挖掘方法无法用于数据流[4]。挖掘数据流即从连续不间断的流数据中提取隐藏的知识/模式的过程,如图1所示。数据流挖掘包括数据流分类、数据流聚类和数据流上的关联规则挖掘等[5-7]。其中,数据流聚类是将数据对象集合中的相似对象划分为一个或多个"簇"的过程[8-18],划分后同一簇中元素彼此相似,不同簇中元素彼此相异。

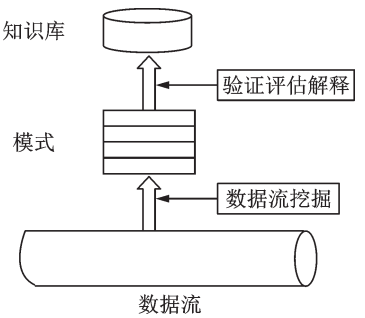

图1 数据流挖掘过程

Fig.1 Data stream mining process

不同于传统静态数据聚类问题,数据流聚类因数据本身的特性造成了诸多限制,影响了传统算法的直接使用。例如:森林中安放了数千个传感器,气象站从所有传感器连续不断、高速地接收有关温度、风速、方向、湿度和传感器位置等天气状况信息。由于数据流是无界且不断发展的,采用传统聚类算法进行批处理不可行。同时,它也无法全部存储在内存中,而是需要增量存储并对数据进行快速处理。此外,现实场景中,传感器暴露在各种不同的天气条件下,极有可能出现故障,如因电池电量不足、无网络连接或火灾引起的数据缺失或者数据异常。聚类算法应能随着时间的推移提供有效聚类,并在出现异常需要采取行动时"突出显示"这些异常值,如指示更换传感器、灭火等。故本文认为一个好的数据流聚类算法可应对以下挑战:(1)简洁表示已发现的簇;(2)增量式且快速处理新到数据;(3)可快速检测孤立点。

近年来,越来越多的数据流聚类算法被提出,且由于设计出的算法种类繁多,为了方便后续的追踪与研究,很多学者从不同角度对提出的算法进行综述[6,8,12-15]。文献[6]研究了电网问题的一个实例。由电网络传播的大约4 000个传感器连续生成数据流,对电子负载进行每小时/每日/每周预测,使用PiD算法进行数据预处理[19],作用ODAC聚类算法进行内部相关度量[20],以及用VFDT进行决策树分类[21]。虽然该研究给出了数据流挖掘的实用性,但缺乏深入的分析,且其使用的算法已经过时。文献[22]讨论了更多的挖掘任务和实际应用,但仍然没有对当前的数据流挖掘方法有足够的总结。文献[8]描述了数据流聚类在网络入侵检测等领域的应用。文献[12]对2016年以前的经典数据流聚类进行了分析。文献[13]聚焦于数据流聚类算法在大数据集上的数据分析。文献[14]对2017年提出的数据流算法进行分类总结。文献[15]对基于距离的数据流聚类算法进行详细综述。不同于以上综述,本文引入数据流聚类算法的一般框架,并给出有依据的分类指标,对2002年至今最流行的数据流聚类算法进行分类分析,同时给出数据流聚类的软件和工具以及常用数据集,为未来算法设计提供基础。

1 问题描述

1.1 数据流聚类

设数据流$\mathcal{DS}=\{x_1,x_2,\cdots,x_n\}$($n$的取值可以无限大)为1个带有时间戳的多维数据点集合,其中每个数据点$x_i=(x_i^1,x_i^2,\cdots,x_i^d)$是1个包含$d$维的数据记录,其到达时间为$t_i$。与传统静态数据聚类相比数据流聚类具有以下不同。

(1)一遍扫描:数据的实时到达产生了巨大的数据量,受限于存储设备的大小和算法的时间复杂度问题,数据流聚类应是单次扫描,按照流入顺序依次读取数据元素,而传统静态数据聚类可以扫描数据多次。

(2)实时响应:数据流聚类的多数应用要求连续在线的挖掘且具有较短响应时间,而传统静态数据聚类的时间可以无限。

(3)有限内存:数据流中的数据具有海量特征,内存及硬盘无法存储整个数据流集,而传统数据是静态的,数据量一般相对较小,可存储。

(4)概念漂移检测:数据流聚类过程中,因数据分布可能随时间的推移而发生变化,故产生概念漂移。概念和概念漂移的定义如下。

定义1 概念。产生数据流的过程可以被考虑为在随机变量 Y 和 $X=\{X_1,\cdots,X_n\}$ 上的联合分布,这里 $y\in\mathrm{dom}(Y)$ 表示类别标签,$x_i\in\mathrm{dom}(X_i)$ 表示属性值,其中 $\mathrm{dom}(\cdot)$ 表示随机变量的域。

数据流背景下,概念随时间变化,t 时刻的概念可表示为

$$P_t(X,Y) \tag{1}$$

定义2 概念漂移。当 t 时刻和 u 时刻的分布发生变化,则概念漂移发生,有

$$P_t(X,Y)\neq P_u(X,Y) \tag{2}$$

(5)传统数据聚类的结果通常是精确的,但数据流聚类一般只能得到近似结果。

为了更有效了解数据流聚类过程,图2给出一个通用数据流聚类框架。当数据流到达,使用缓冲区存储最新数据。随后使用流聚类引擎读取缓冲区中数据创建内存中数据的概要。为了持续获得概要,流聚类引擎可以应用不同的时间窗口和计算方法。接着用户请求被触发,流聚类引擎对概要进行处理并输出近似结果。其中大多数数据流聚类算法都基于传统聚类算法。最后应用流聚类评估方法对数据流聚类算法的性能进行评估。

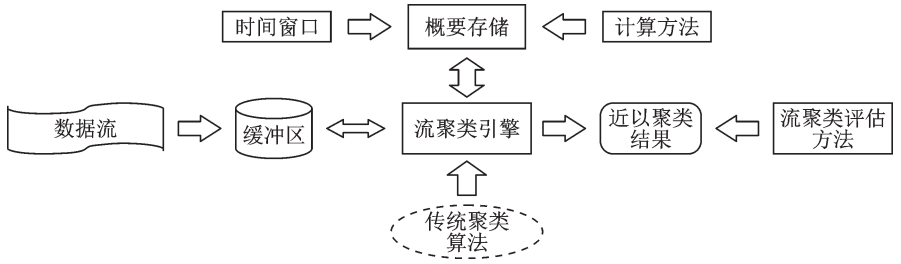

图2 数据流聚类算法的一般框架

Fig.2 General frame of data stream clustering algorithm

1.2 时间窗口

一般来说,数据流是无限的,所以某个时刻只能处理流的一部分,这里定义为时间窗口,表示为 $W[i,j]=(x_i,x_{i+1},\cdots,x_j)$,其中 $i<j$。时间窗口旨在"忘记"旧数据,以避免历史数据使分析偏向于过时的模式。时间窗口模型主要有4种:界标窗口、滑动窗口、衰减窗口和倾斜时间窗口,如图3所示。

(1)界标窗口

界标窗口根据事件将数据流分割成不连贯的块。从开始时刻1到当前时刻 t_c 的数据,记为 $W[1,t_c]$,即界标到达后的所有数据点都同样重要,没有过去和现在之间的差异。每当出现新界标时,删除窗口中的所有数据,并捕获新数据。但是,随着数据流的不断发展,新数据点可能与之前旧数据点构建的模型出现不一致,故可使用其他3种窗口模型强调最近数据。

(2)滑动窗口

滑动窗口只对最近的 w 个数据点感兴趣,用

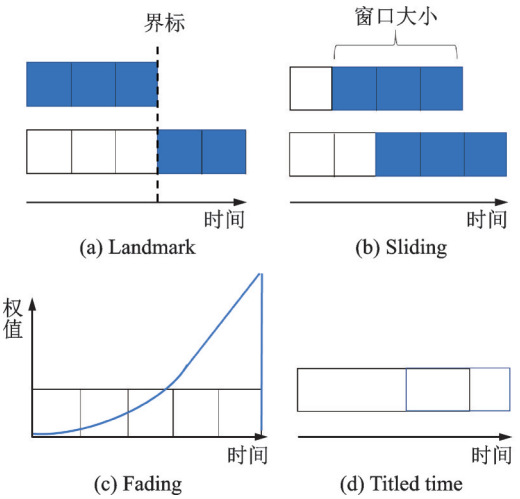

图3 时间窗口

Fig.3 Time window

$W[t_c-w+1,t_c]$表示。其基于先进先出原则,即一旦有新数据点可用,窗口中最老的数据点就会被删除,这样减少了对内存的需求。但挖掘结果依赖窗口大小w,如果w太大,可能出现概念漂移,并由于窗口中包含了太多的过时信息,导致模型的精度下降。如果w太小,窗口可能因数据贫乏而导致模型过拟合或者大的方差。之前的工作使用了固定的窗口大小w,而最近的工作开始使用可伸缩的窗口大小w,当模型精度较高时,放大w的值,当模型精度较低时,缩小w的值。

(3)衰减窗口

衰减窗口强调近期数据的重要性,衰减历史数据对计算结果的影响。每个数据元素对最终结果的影响用权值来表示,并随着时间的推移而逐渐减小。每次迭代中,权重通过1个因子(如$2^{-\lambda}$)衰减,其中衰减因子λ影响衰减速率。由于每次迭代中衰减权值的计算成本很高,所以权值可以在固定的时间间隔内更新,也可以在更新簇时更新。如衰减函数可表示为$w(\Delta t)=2^{-\lambda\Delta t}$,其中$\Delta t$表示聚簇上次更新的时间。

(4)倾斜时间窗口

倾斜时间窗口可在不同级别时间粒度层上进行分析和挖掘。通常人们感兴趣的是细粒度层上的当前数据,而不是粗粒度层上的长期数据。为此,倾斜窗口计算中,在最细粒度层上记录和运算最近数据,在较粗粒度层上记录和计算较久远数据。倾斜窗口在存储空间和精度上提供了良好的权衡,近似存储了整个数据集。但随着长时间运行,模型稳定性仍可能下降。

1.3 计算方法

数据流聚类通常采用以下两种计算方法。

(1)增量学习。增量学习方法中,模型逐渐发展以适应传入数据的更改。增量学习方法如图4(a)所示,可采用按数据实例和窗口两种方案进行更新,具有即时提供挖掘结果的优势,但需要更多的计算资源。

(2)两阶段学习。两阶段学习,也称为在线-离线学习。第一阶段(在线阶段),采用实时方式更新数据概要。第二阶段(离线阶段),当用户发送请求时基于已存储的概要进行挖掘。两阶段学习方法如图4(b)所示。这种方法能够处理数据高速流,但是它的局限性在于用户必须等到挖掘结果。

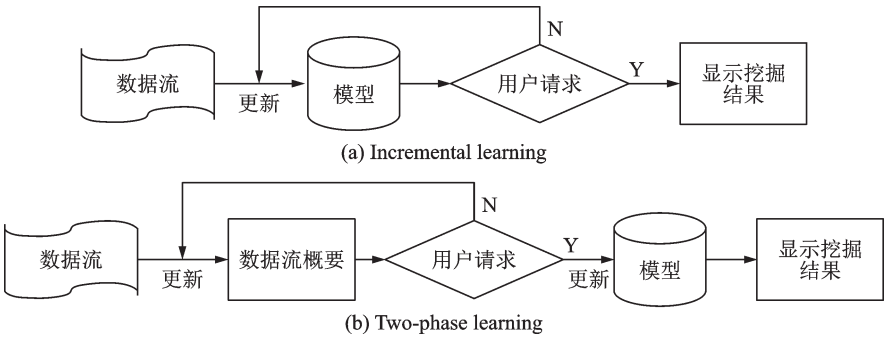

图4 计算方法

Fig.4 Learning method

1.4 流聚类评估方法

为了对各种数据流聚类算法的精度进行评估,常用以下3项评价指标[23]:精确度(Accuracy, Acc);归一化互信息(Normalized mutual information, NMI);兰德指数(Rand index, RI)。

(1) 精确度Acc[24]

精确度Acc计算公式如下

$$\mathrm{Acc}=\frac{\sum_{i=1}^{K}\frac{|C_i^d|}{|C_i|}}{K}\times 100\% \tag{3}$$

式中:K表示聚簇个数;$|C_i^d|$表示在聚簇i中的样本点数;$|C_i|$表示聚簇i中真实的样本个数。因此,Acc度量了聚簇的精度,Acc∈[0,1],Acc越大表明聚类精度越高。

(2) 归一化互信息[25]

归一化互信息NMI是量化两个分布之间共享统计信息的对称策略。当聚簇标签和真实样本类别一对一映射时,NMI值得到最大值1。给定真实聚簇$A=\{A_1,A_2,\cdots,A_k\}$和某聚类算法得到的聚簇$B=\{B_1,B_2,\cdots,B_h\}$,混淆矩阵$\boldsymbol{C}$中的元素C_{ij}表示既在A_i又在B_j中的样本个数。NMI计算如下

$$\mathrm{NMI}(A,B)=\frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B}C_{ij}\log(C_{ij}N/C_{i.}C_{.j})}{\sum_{i=1}^{C_A}C_{i.}\log(C_{i.}/N)+\sum_{j=1}^{C_B}C_{j}\log(C_{j}/N)} \tag{4}$$

式中:$C_A(C_B)$表示$A(B)$中样本个数;$\boldsymbol{C}_{i.}(\boldsymbol{C}_j)$表示$C$中$i$行元素和;$N$表示样本个数。

(3) 兰德指数[26]

RI比较$n\times(n-1)/2$个数据对,其中n为数据集中样本个数,P_1、P_2表示两种聚类算法,n_1为数据对(x_i,x_j)在P_1、P_2中划分为同一类的数据对数,n_{00}则为(x_i,x_j)隶属不同类的数据对数,RI计算如下

$$\mathrm{RI}=\frac{n_{11}+n_{00}}{C_n^2} \tag{5}$$

由式(5)可得RI∈[0,1], RI=1表示P_1与P_2划分完全一致。

2 数据流聚类算法

现有数据流聚类算法一般由传统聚类算法扩展而来,可将其根据所扩展的传统算法分为5类:基于划分的方法;基于层次的方法;基于密度的方法;基于网格的方法;基于模型的方法。表1分别针对如下特性对现有方法进行总结:(1)基算法;(2)时间窗口;(3)计算方法;(4)聚簇个数是否自适应;(5)是否可挖掘拓扑结构;(6)是否可检测概念漂移;(7)是否适合高维数据。如表1所示,基于划分的数据流聚类方法易于实现且相对简单,但需预定义聚簇的个数。例如,STREAM算法[27]应用K-median算法,使用中位数来计算中心,将聚类问题简化为数据点到最近聚簇的距离代价最小化问题,即找出代价最低的聚簇的数量和位置。文献[28]基于K-Means算法聚类数据流,克服了基于划分的聚类算法需要给出聚簇个数k及适应概念漂移这两个问题。FEAC-Stream算法[29]基于K-Means算法,使用进化算法自动估计聚簇个数k。完全在线时,FEAC-Stream不存储数据概要,而是维护最终的聚类结果;运行时使用Page-Hinkley测试跟踪聚类质量,如果质量下降,算法自行调整。

基于层次的数据流聚类方法虽可发现有意义的聚簇结构,但其具有较高的计算代价,而且对流数据到达的顺序敏感。例如,CluStream算法[30]允许在不同的时间范围而不是整个数据流上执行聚类。它还通过存储所有时间戳的线性和以及平方和来扩展聚类特征(Clustering feature vector, CF)。为了支持不同的时间范围,算法按照金字塔方案定期存储当前微聚簇的快照。虽然有些快照会定期更新,但为了维护关于历史数据的信息,有些快照更新得不那么频繁。通过从存储的快照中减去当前的微聚簇,可以近似地得到数据流的期望部分。然后,将提取的微聚簇运行K-Means的变体算法来生成宏观

表1 数据流聚类算法比较

Table 1 Comparison of various data stream clustering algorithms

算法	提出年份	分类	基算法	时间窗口	增量学习	自适应聚簇个数	发掘拓扑结构	检测概念漂移	适合高维数据
STREAM[27]	2002	基于划分方法	K-medians	界标窗口	√	×	×	×	×
Adaptive Streaming K-Means[28]	2016	基于划分方法	K-Means	滑动窗口	√	√	×	√	√
FEAC-Stream[29]	2017	基于划分方法	K-Means	衰减时间窗口	√	√	×	√	√
CluStream[30]	2003	基于层次方法	BIRCH	倾斜时间窗口	×	√	×	√	×
HPStream[11]	2004	基于层次方法	BIRCH	倾斜时间窗口	×	√	×	√	√
SWClustering[31]	2008	基于层次方法	BIRCH	倾斜时间窗口	×	√	×	√	×
E-Stream[32]	2007	基于层次方法	BIRCH	倾斜时间窗口	×	√	×	√	×
REPSTREAM[33]	2009	基于层次方法	CHAMELEON	衰减时间窗口	√	√	×	√	×
DenStream[24]	2006	基于密度方法	DBSCAN	衰减时间窗口	×	√	×	√	×
ACSC[34]	2019	基于密度方法	DBSCAN	衰减时间窗口	√	√	×	√	×
OPTICS-Stream[35]	2007	基于密度方法	OPTICS	衰减时间窗口	×	√	×	√	×
incPre-Decon[36]	2011	基于密度方法	PreDecon	界标窗口	√	√	×	×	√
HDDStream[37]	2012	基于密度方法	DBSCAN	衰减时间窗口	×	√	×	√	√
MuDi-Stream[38]	2013	基于密度方法	DBSCAN	衰减时间窗口	×	√	×	√	√
CEDAS[39]	2017	基于密度方法	DBSCAN	衰减时间窗口	√	√	√	√	√
Impoved Data Stream Clustering[40]	2018	基于密度方法	DBSCAN	衰减时间窗口	×	√	×	√	√
I-HASTREAM[41]	2016	基于密度方法	OPTICS	衰减时间窗口	×	√	×	√	√
D-Stream[42]	2007	基于网格方法	DENCLUE	衰减时间窗口	×	√	×	√	×
MR-Stream[43]	2009	基于网格方法	STING	衰减时间窗口	×	√	×	√	×
CellTree[44]	2004	基于网格方法	STING	衰减时间窗口	×	√	×	√	×
SWEM[45]	2009	基于模型方法	EM	衰减时间窗口	√	√	×	√	×
GCPSOM[46]	2009	基于模型方法	SOM	衰减时间窗口	√	√	√	√	×
SVStream[47]	2011	基于模型方法	SVM	滑动窗口	√	√	×	√	×
G-Stream[23]	2016	基于模型方法	GNG	衰减时间窗口	√	√	√	√	×
RPGStream[48]	2020	基于模型方法	GNG	衰减时间窗口	√	√	√	√	√

簇。基于CluStream算法框架,许多改进算法被提出。HPStream[11]引入投影聚类,为每个聚簇(子空间聚簇)选择最佳属性集,获得了比CluStream更好的聚类质量。SWClustering算法[31]为滑动窗口创建了时间聚簇特征解决了当微聚簇中心逐渐移动时,CluStream以不断增长的半径维持该微聚簇,而未将其分裂成许多更小的微聚簇导致的聚类性能下降,在运行时间和内存使用方面具有更好的性能。E-Stream算法[32]将聚簇演化分为5个类型:出现、消失、自演化、合并和分裂,并使用衰减模型和簇直方图来识别簇进化的类型。受CHAMELEON算法启发,REPSTREAM算法[33]是一种基于图的数据流层次聚类方法。为了识别聚簇,REPSTREAM更新2个稀疏图,这2个稀疏图是通过连接每个顶点到它的K-近邻顶点形成的。第1个图捕获了到达数据点之间的连接关系,用于选择1组代表性的顶点。

第2个图由具有代表性的顶点构成,有助于在更高层次上做出聚类决策。REPSTREAM应用衰减窗口减少旧数据的影响。REPSTREAM跟踪代表性顶点之间的连接性,并根据它们的连接性进行合并或拆分。

基于密度的数据流聚类方法可发现任意形状的聚簇,但算法预设参数太多是它的弊端。例如,DenStream算法[24]与CluStream一样使用微聚簇捕获数据流的概要信息。在线阶段每个微聚簇都具有一个由聚类特征向量得到中心和半径。微聚簇可分为3种类型:核心微聚簇、潜在微聚簇和孤立点微聚簇。离线阶段,在3种类型的微聚簇上应用DBSCAN算法。ACSC蚁群流聚类算法[34]提供一种对非平稳聚簇进行总结的流聚类方法,同时解决聚簇个数变化问题。使用采样解决数据流聚类的速度要求,点与聚簇的相似性评估仅使用这个聚簇中的一个点。随机抽样方法取代了传统的穷举搜索每个点合适的微聚簇,然后搜索每个微聚簇的最近邻,使用蚂蚁启发的排序方法对这些聚簇进行细化。OPTICS-Stream算法[35]利用OPTICS算法中的核心距离和可达距离进行聚类。incPre-Decon算法[36]支持单个数据对象更新和块更新,可用于处理动态数据流。HDDSTREAM算法[37]是一个基于密度的高维数据流投影聚类算法。HDDSTREAM算法提出了一种总结结构,即投影微聚簇,用于在相关维度上总结一组对象。在流环境中,聚簇很可能会随着时间的改变成为离群点,故其提出了不同类型的微聚簇,以便逐步形成真实的投影微聚簇并安全去除离群点。与HPStream相反,当1个新数据到来,如果它远离所有现有聚簇,则将成为1个新聚簇的种子(即使它是1个离群点),一些旧聚簇将被删除,以保持聚簇的总数量不变。MuDi-Stream[38]是一种在线-离线数据流聚类算法。在线阶段,以核心微聚簇形式保存多密度数据流演进的汇总信息;离线阶段使用自适应的基于密度的聚类算法生成最终的聚簇。基于网格的方法作为离群值缓冲来处理噪声和多密度数据,并减少聚簇合并时间。CEDAS算法[39]可聚类演变的概念漂移数据流到任意形状的簇。文献[40]提出一种基于密度的两阶段算法,适用于任意形状的聚簇,其主要特点是根据输入的数据自动调整阈值。I-HASTREAM算法[41]也是基于密度的两阶段算法,是HASTREAM算法的改进版本。在线阶段,它为数据概要创建为微聚簇;离线阶段,微聚簇得到最小生成树,最后采用层次聚类进行聚类。

基于网格的数据流聚类方法不仅运行速度较快,也可发现任意形状聚簇,但其聚类质量取决于选取的网格粒度。例如,D-Stream算法[42]采用两阶段学习。在线阶段中将输入数据点映射到网格,离线阶段中计算网格密度并基于密度对网格进行聚类。与其他算法不同,D-Stream自动、动态地调整聚簇,不需要用户指定目标时间范围和聚簇个数。MR-Stream算法[43]通过使用单元网格(可以使用树状数据结构动态地细分为更多单元)来促进在多种分辨率下发现聚簇。在线阶段,它将新传入数据分配给适当的单元,并更新概要信息;离线阶段在固定高度h获得树的一部分,并在h确定的分辨率级别上执行聚类。CellTree算法[44]首先将数据空间划分为一组互斥的大小相同的单元格。当单元格的权重大于某一阈值时,采用混合划分方法即对μ-划分和σ-划分进行折中,将单元格动态划分为2个中间单元。μ-划分通过最大化标准差选择维度,而σ-划分选择标准差最小的维度。

基于模型的数据流聚类算法包含了很多领域知识,强依赖于假设模型,例如:SWEM算法[45]基于EM模型,GCPSOM算法[46]基于SOM模型,SVStream算法[47]基于SVM模型,G-Stream[23]和RPG-Stream[41]算法基于生长型神经气(Growing neural gas, GNG)模型。表2给出了各类算法的优缺点比较。图5突出显示了算法之间的关系,并显示了随着时间的推移,概念是如何被完善和改进的。

2.1 Adaptive Streaming K-Means算法

Adaptive Streaming K-Means通过扩展传统基于划分的K-Means算法聚类数据流。通常,基于划分的聚类算法需要给出聚簇个数k,且难以适应输入数据中的概念漂移。但Adaptive Streaming K-Means

表2 数据流聚类算法优缺点比较

Table 2 Advantages and disadvantages of partial data stream clustering algorithms

算法	优点	缺点
基于划分方法	简单且易实现	①需要预设聚簇个数k;②只能发现球形聚簇
基于层次方法	得到更有意义的聚簇结构	①高复杂性;②对输入数据点顺序敏感
基于密度方法	可发现任意形状聚簇且对噪声鲁棒	①参数个数多;②难以检测到不同密度的簇
基于网格方法	可快速发现任意形状聚簇且对噪声鲁棒	①聚类质量强依赖于网格粒度;②不适用于高维数据流聚类
基于模型方法	简单且包含领域知识	强依赖于模型

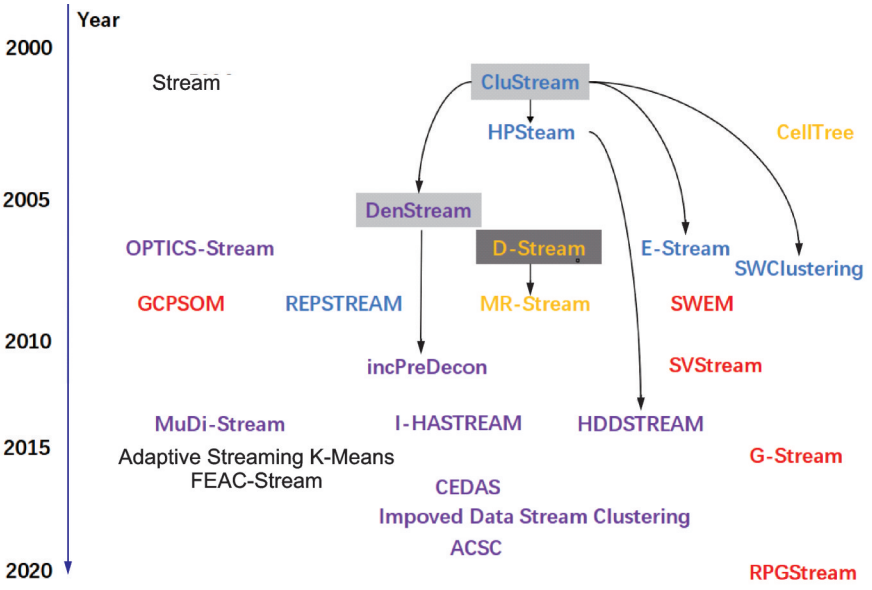

图5 数据流聚类算法发展

Fig.5 Development of stream clustering algorithms

克服了这两个问题。

Adaptive Streaming K-Means算法主要分为2个阶段,即初始化阶段和连续聚类阶段。在初始化阶段,l个数据点被累积,然后确定候选中心。为了找到聚簇个数k并确定候选中心,通过使用核密度估计(Kernel density estimation, KDE)计算数据的概率密度函数(Probability density function, PDF)。PDF曲线形状的所有方向变化都被认为是新区域开始的标志。区域可以定义为PDF曲线两个连续方向变化之间的区域,将区域数量作为候选k,区域的中心作为候选初始中心。由于不同的特征通常表现出不同的分布,k值超过1个,因此会找到不同的候选中心。对$k\in[k_{\min},k_{\min}+k_{\max}]$进行聚类,根据轮廓系数比较不同$k$值的聚类结果,选择最好的$k$以及与其对应的中心。在连续聚类阶段,先执行概念漂移检查,如果没有发生概念漂移,则对输入数据进行聚类。如果存在概念漂移,则重新计算k和中心(重新初始化算法),然后继续使用新的k和中心进行聚类。对于概念漂移检测,算法执行过程中存储输入数据的标准差和均值。跟踪这2个值如何随时间变化,并根据变化预测概念漂移。当1个概念漂移被预测时,当前的聚类中心不再有效,并触发重新初始化。使用这种机制,算法捕获概念漂移并适应输入流。鉴于算法本质为K-Means,该算法仍有只能检测到超球形簇的缺陷。

2.2 CluStream算法

CluStream算法通过扩展传统聚类方法BIRCH聚类数据流，使用微聚簇来构造数据流的总结信息。该算法扩展了BIRCH的CF，允许在不同时间范围而不是整个数据流上执行聚类。根据扩展CF定义微聚簇如下。

定义3 微聚簇。带时间戳$T_{i_1},\cdots,T_{i_n}$的d维数据点集$X_{i_1},\cdots,X_{i_n}$，其中$X_{i_j}=(x_{i_j}^1,x_{i_j}^2,\cdots,x_{i_j}^d)$。微聚簇被定义为$(2\cdot d+3)$元组$(\mathbf{CF2}^{\mathrm{x}},\mathbf{CF1}^{\mathrm{x}},\mathrm{CF2}^{\mathrm{t}},\mathrm{CF1}^{\mathrm{t}},n)$。其中，矢量$\mathbf{CF2}^{\mathrm{x}}$和$\mathbf{CF1}^{\mathrm{x}}$分别表示每维数据值的平方和及累加和，如第$p$维的$\mathbf{CF2}^{\mathrm{x}}$和$\mathbf{CF1}^{\mathrm{x}}$分别表示为$\sum_{j=1}^{n}(x_{i_j}^p)^2$和$\sum_{j=1}^{n}(x_{i_j}^p)$；标量$\mathrm{CF2}^{\mathrm{t}}$和$\mathrm{CF1}^{\mathrm{t}}$分别表示时间戳的平方和及累加和；$n$是数据点的个数。

CluStream在线阶段通过收集数据并使用K-Means算法创建q个簇进行初始化。当新的数据点到达，它会被离它最近的微聚簇所吸收，否则创建新的聚簇。为了保持微聚簇的数量不变，过时的微聚簇会根据其平均时间戳的阈值被移除或者合并两个最近的微聚簇。为支持不同的时间粒度，该算法应用倾斜时间窗口定期存储当前CF的快照。虽然有些快照会定期更新，但有些快照为维护历史数据的信息而更新不频繁。通过从先前CF的存储快照中减去当前CF，可以近似得到流的所需部分。离线阶段通过提取的微聚簇运行K-Means变体算法生成宏聚簇。CluStream虽可产生高质量的聚类结果，但不能用于高维数据流聚类。

2.3 DenStream算法

DenStream算法使用基于CF向量的特征向量，通过创建两种类型的微聚簇（潜在微聚簇和孤立点微聚簇）克服了CluStream算法对噪声的敏感性。

在线阶段每个潜在微聚簇都具有1个相关的权值w表示其基于时间的重要性。通过衰减函数$f(t)=2^{-\lambda t}(\lambda>0)$，每个数据点的权值随时间$t$呈指数衰减。如果权值$w=\sum_{j=1}^{n}f(t-T_{ij})$大于阈值$\mu$，则将该聚簇视为核心微聚簇，其中$T_{i1},\cdots,T_{in}$是数据点$p_{i1},\cdots,p_{in}$的时间戳。$t$时刻，如果$w\geqslant\beta\mu$，则聚簇是潜在微聚簇，否则它是孤立点微聚簇，其中$\beta$是孤立点相对于核心微聚簇的阈值$(0<\beta<1)$。当1个新的数据点到达，DenStream根据其更新的半径将其插入到最近的潜在微聚簇中。如果插入失败，尝试将该数据点插入到最近的孤立点微聚簇中；如果插入成功，更新聚簇概要统计信息。否则，创建1个新的孤立点微聚簇吸收这个点。DenStream采用剪枝在孤立点缓冲区检查孤立点微聚簇的权重，以保证能够识别出真正的孤立点。然而，当删除1个微聚簇或合并2个旧的微聚簇时，不释放分配的内存单元，以及删除孤立点的剪枝过程相当耗时，这是DenStream算法的不足。离线阶段将在线阶段发现的潜在微聚簇传递到DBSCAN算法，以确定最终的聚类结果。

2.4 ACSC算法

ACSC蚁群流聚类算法[34]提供一种对非平稳聚簇进行总结的流聚类方法，同时解决聚簇个数变化问题。该算法使用采样解决数据流聚类的速度要求，点与聚簇的相似性评估仅使用这个聚簇中的1个点。同时，随机抽样方法取代了传统的穷举搜索每个点合适的微聚簇，然后搜索每个微聚簇的最近邻。单遍数据扫描后创建粗粒度的聚簇，第1个数据点即为第1个聚簇，后面的数据点被分配到1个相似的现有聚簇，如果与所有聚簇都不相似，则生成1个新的聚簇。只有在每个点被分配到各自的聚簇之后，才会创建微聚簇。每个点被转换为1个微聚簇，这些微聚簇只尝试与同一聚簇中的其他点合并，合并操作代价高。

因为单遍扫描后生成的聚簇通常粗糙且个数多，使用蚂蚁启发的排序方法对这些聚簇进行细化。

这种方法是建立在蚂蚁行为上的模型,蚂蚁会把尸体聚集到“墓地”或分类,即孤立的数据点应该被捡起来,然后丢在其他有类似数据点的地方。排序蚂蚁被分配到每个聚簇中,它们试图通过概率性地挑选微聚簇并将它们放到更合适的聚簇中来精炼初始聚簇。这些操作倾向于解散较小的聚簇,将其内容移动到相似的、较大的聚簇。直观地说,聚簇中所有数据点应放在一个大的聚簇中,而不是放在许多分布的小聚簇中。DenStream算法中微聚簇由2个参数定义:最大半径ε和最小值。最小值表示ε内密集的微聚簇的最小数据点数。但ACSC中每个点最初被视为自己的微聚簇,因此参数最小值为1,故不再需要。这使得将微聚簇定义为核心、潜在或离群值的复杂性降低,从而每个微聚簇都被平等地对待。为进一步简化,ACSC将密度可达、直接密度可达和密度连接的概念合并为仅密度可达1个概念,其决定2个微聚簇是否连接,并被视为同一聚簇的不同部分。ACSC在创建和合并微聚簇之前会给聚簇分配样本点,因此当1个新的微聚簇直接密度可达时,它也会密度可达并密度连接到聚簇中的所有微聚簇。这样降低了算法整体的复杂性,并允许有效采样。ACSC有如下改进:(1)将总结和聚类两个阶段结合成1个在线过程;(2)微聚簇使用单个参数ε定义,总体只需要3个参数;(3)使用单个密度概念形成聚簇,将离群点识别为包含单个数据点的聚类;(4)通过对每个聚簇进行采样,本地进行排序操作,减少了计算时间。

2.5 D-Stream算法

D-Stream算法结合了基于密度和网格的方法。在线阶段将接受到的每个数据点映射到某个网格中,离线阶段计算网格密度,并基于密度将网格进行聚类。D-Stream算法进行了下列改进:

(1)为了获得簇的动态演变,提出了与每个数据点密度相关的密度衰减因子。衰减因子赋予近期数据更大的权重,并且没有丢弃历史信息。

(2)由于数据流海量,完全保存每个数据记录的信息不可行,D-Stream算法将数据空间划分成网格单元,并将数据点映射到对应的网格单元中,同时更新网格的特征向量。使用网格作为概要数据结构,不需要对原数据进行保存只需要对网格单元进行操作,具有较快的处理时间。

(3)采用基于密度的方法对网格单元进行聚类,可以发现任意形状的簇,能有效解决CluStream算法等只能识别球形簇的问题。

(4)可动态调整簇,且不需要用户指定目标时间范围和聚簇的个数。

D-Stream算法尽管存在多种优点,同时也存在一些不足之处,如:采用网格概要数据结构会丢失数据的位置信息;当被应用于聚类高维数据流时,网格单元的个数可能会很大;对数据空间进行均匀网格划分,网格的划分粒度将直接影响聚类的精度。使用基于密度的计算方法对当前的数据流聚类能产生很多的聚类结果,不利于对历史信息查询。算法没有对低密度的网格单元进行处理,也会影响聚类的精度。

2.6 RPGStream算法

RPGStream算法基于随机投影理论,可用于解决高维大数据流聚类(n和d均很大)。

定义4 随机投影。原始的d维数据使用随机矩阵$\boldsymbol{R}_{d\times d_c}$被投影到$d_c$($d_c \ll d$)维子空间。样本矩阵$\boldsymbol{X}_{n\times d}$($n$个样本,$d$维)利用式(6)投影到$\boldsymbol{X}^{RP}_{n\times d_c}$,即

$$\boldsymbol{X}^{RP}_{n\times d_c}=\boldsymbol{X}_{n\times d}\boldsymbol{R}_{d\times d_c} \tag{6}$$

JL引理[37]是随机投影的理论依据,即高维欧氏空间的点映射到低维空间,相对距离可得到一定误差范围内的保持。

定理1[37] 设样本矩阵$\boldsymbol{X}\in \mathbf{R}^{n\times d}$包含$n$个样本$d$维特征,给定$\varepsilon,\beta>0$,则

$$k_0=\frac{4+2\beta}{\varepsilon^2/2-\varepsilon^3/3}\log n \tag{7}$$

式中:参数ε控制距离保持的精度;β控制投影成功的概率。随机矩阵$\boldsymbol{R}$为一个$d\times d_c$矩阵,d_c为正整数,且$d_c\geqslant k_0$,设$E=(1/\sqrt{d_c})\boldsymbol{XR}$,$f:\boldsymbol{R}^d\rightarrow\boldsymbol{R}^{d_c}$将$\boldsymbol{X}$的第$i$行映射到$E$的第$i$行。

对所有的$u,v\in\boldsymbol{X}$,在至少$1-n^{-\beta}$概率下,有

$$(1-\varepsilon)\|u-v\|^2\leqslant\|f(u)-f(v)\|^2\leqslant(1+\varepsilon)\|u-v\|^2 \tag{8}$$

从式(8)看出,理论上JL界(k_0)不依赖于原始空间的维度d,为得到定理1的结果,只需生成随机投影矩阵$\boldsymbol{R}$并进行投影计算。随机投影矩阵$\boldsymbol{R}(i,j)=r_{ij}$,$r_{ij}$为一个独立的随机变量,可以由以下3种概率分布生成

$$r_{ij}\sim N(0,1) \tag{9}$$

$$r_{ij}=\begin{cases}+1 & \text{with probability } 1/2\\ -1 & \text{with probability } 1/2\end{cases} \tag{10}$$

$$r_{ij}=\sqrt{3}\times\begin{cases}+1 & \text{with probability } 1/6\\ 0 & \text{with probability } 2/3\\ -1 & \text{with probability } 1/6\end{cases} \tag{11}$$

RPGStream算法首先依据d_c的大小生成随机投影矩阵$\boldsymbol{R}_{d\times d_c}$,当一个新的数据点$x_i$到达时,用$\boldsymbol{R}_{d\times d_c}$将$x_i$投影到对应的$d_c$维($d_c\ll d$)点$y_i$。从而整个数据流$\mathcal{DS}_{n\times d}=\{x_1,x_2,\cdots,x_n\}$被投影到$\mathcal{Y}_{n\times d_c}=\{y_1,y_2,\cdots,y_n\}$,再对$\mathcal{Y}_{n\times d_c}$使用G-Stream算法进行聚类。RPGStream生成了包含一系列神经元节点的结构,每个节点的权值向量为$\boldsymbol{w}_c=(w_c^1,w_c^2,\cdots,w_c^d)$。

3 软件与工具

数据流聚类在实际应用中的一个重要方面是可用的软件和工具。只有少数作者为他们的算法提供参考实现,例如,C、C++或R实现可以用于实现STREAM算法[49]和REPSTREAM算法。

可用的数据流挖掘软件包括:

(1)WEKA①。WEKA是学术领域最知名的数据挖掘软件。WEKA包括数据预处理、回归、分类、聚类和关联规则等1组学习算法。

(2)大规模在线分析(Massive online analysis, MOA)②[50]。MOA是用于数据流挖掘流行的开源框架,它包含了分类和聚类的离线在线集合,以及用于评估的工具。实现了流聚类算法D-Stream, DenStream和CluStream。

(3)RapidMiner③。RapidMiner是另一个用于数据挖掘的开源软件。RapidMiner比WEKA更强大,因为它包含了WEKA中的所有算法和其他高级算法,并且将挖掘过程定义为一系列操作人员,提供更多可视化工具,更加直观。

(4)streamMOA。为了更快地创建原型,还存在统计编程语言R的流包[51]。它包含处理数据流的通用方法,还实现了D-Stream等算法。扩展包streamMOA[52]接口对DenStream和CluStream算法进行MOA实现。为了在高维空间中处理数据,子空间MOA框架[53]提供了HDDStream的Java实现。同样,

①http://www.cs.waikato.ac.nz/ml/weka

②https://moa.cms.waikato.ac.nz/

③https://rapidminer.com/

R-语言子空间MOA框架也可接口两种方法,使它们可以被流包访问。

(5)streamDM华为诺亚方舟实验室2015项目[54]提供了Spark流数据挖掘方法,Spark流是Spark引擎的扩展。目前它实现了CluStream等算法,并计划用更多的流聚类算法扩展项目。

4 数据集

为了测试数据流聚类性能,常使用人工数据集。人工数据集让用户有机会指定流的属性,如噪声比、概念漂移、聚簇形状和密度等。MOA有许多类不同的MOA流生成器,可生成不同形状、有或没有概念漂移的人工数据流。除了人工数据集,其他流行的真实数据集如表3所示。

表3 流行的真实数据集

Table 3 Statistics of popular datasets

数据集	网址	样本数	特征维数	类别数
Hyper Plane Stream	http://www.cse.fau.edu/~xqzhu/stream.html	100 000	10	5
Network Intrusion Detection Subset KddCup99	http://archive.ics.uci.edu/ml/datasets/KDD+Cup+1999+Data	494 021	41	23
Forest Cover Type CoverType	https://archive.ics.uci.edu/ml/datasets/Covertype	581 012	54	7
The Daily and Sports Activities Data Set ACT2	http://archive.ics.uci.edu/ml/datasets/Daily+and+Sports+Activities	9 120	5 625	19
Sensor Stream	https://www.cse.fau.edu/~xqzhu/stream.html	2 219 803	5	54
Power Supply Stream	https://www.cse.fau.edu/~xqzhu/stream.html	29 928	2	24

5 结束语

本文系统地介绍了数据流挖掘的挑战,给出了数据流聚类的一般框架,并描述了其与传统静态数据聚类之间的关联,对数据流挖掘中的各种聚类算法进行了综述。尽管已有大量数据流聚类算法被提出,数据流聚类依然是极富挑战的研究热点。随着愈加广泛的应用领域中大量数据流的产生,越来越多的研究主题不断涌现出来。

(1)隐私保护和保密。数据流对数据挖掘中的隐私和机密性保护提出了新的挑战和机遇。例如,健康监测数据流、GPS数据流等多隐含用户的生理特征信息、位置信息,一旦数据遭暴露或滥用,严重影响用户的隐私安全。因此,设计基于隐私保护的数据流聚类模型是一个可行且必要的研究方向。

(2)跟踪聚簇演变。数据流中的数据类型具有异质特性(文本、视频、音频、静态图像等),数据处理能力具有差异性(结构化、半结构化、非结构化数据),以多视图视角同时处理这些数据,也将是一类极具挑战性的研究场景。在该类场景下,数据流聚类的一个有趣的未来应用是社会网络分析。社交网络成员的活动可以看作是一个数据流,聚类算法可以用来显示成员之间的相似性,以及这些相似的概况(聚簇)是如何随着时间的推移而演变的。通过跟踪聚簇的演变情况获得有关数据流性质的更多且更有价值的潜在模式。再如,在客户关系管理系统中,管理公司利用数据流聚类算法尝试挖掘新客户群,或者更确切地说是否是现有客户的行为进行了转移。

(3)实时聚类。某些关键的应用领域对反馈时间有着严格的要求。诸如事故检测、航空交通控制等应用领域必须实时给出数据流的聚类结果。虽然目前大多数的数据流聚类算法都已经实现了用户的联机(在线)聚类查询,但这些算法在时间上远远无法满足实时应用的需求。因此,构建快速高效的实时数据流聚类算法亦是十分必要的研究方向。

(4)分布式数据流聚类。现实场景中,数据流数据的采集存在地理位置上的限制,因此分布式数据流聚类算法应运而生。在这类场景下,数据流聚类算法可与MapReduce和分布式计算进行结合,这将加快算法的速度,并处理有限的内存问题。由于现在的存储成本很低,可以仅存储具有代表性的对象(如中心或概要)。通过存储这些代表性对象可使最终用户关注特定的时间段,并通过宏聚类或应用其他数据挖掘算法进一步分析汇总的数据。然而,许多数据聚类技术的并行化并不简单,为了开发一些方法的分布式版本,需要大量的研究和理论分析,以提供新的方法。

(5)数据流概要。如何构建数据流数据信息概要是一个重要的研究方向。到目前为止,数据流聚类最常见的概要是使用微聚簇及其变体。微聚簇通常创建凸形聚簇,而实际上聚簇可以以任何形状出现,比如数据泡,在数据流场景下的"适应"并不那么简单。

(6)数据流算法评估。在数据流环境中,由于存在概念漂移、有限处理时间、验证延迟、多流结构、删失数据等问题,使得单一指标度量动态变化的数据流环境的有效性较为困难。因此,更感兴趣的是算法评估指标如何随着时间的推移而演变。

参 考 文 献:

[1] 朱颖雯, 陈松灿. 基于随机投影的高维数据流聚类[J]. 计算机研究与发展, 2020, 57(8): 1683-1696.
ZHU Yingwen, CHEN Songcan. High dimensional data stream clustering algorithm based on random projection[J]. Journal of Computer Research and Development, 2020, 57(8): 1683-1696.

[2] 朱颖雯, 杨君. 基于欧拉核的数据流聚类算法[J]. 计算机科学, 2019, 46(12): 74-82.
ZHU Yingwen, YANG Jun. Euler kernel-based data stream clustering algorithm[J]. Computer Science, 2019, 46(12): 74-82.

[3] 蒋考林,白玮,任传伦,等. 基于建链信息的密数据流识别方法[J]. 数据采集与处理, 2021, 36(3): 595-604.
JIANG Kaolin, BAI Wei, REN Chuanlun, et al. Identification method of encrypted data flow based on chain building information[J]. Journal of Data Acquisition and Processing, 2021, 36(3): 595-604.

[4] RAMÍREZ-GALLEGO S, KRAWCZYK B, GARCÍA S, et al. A survey on data preprocessing for data stream mining: Current status and future directions[J]. Neurocomputing, 2017, 239: 39-57.

[5] NGUYEN H L, WOON Y K, NG W K. A survey on data stream clustering and classification[J]. Knowledge and Information Systems, 2015, 45(3): 535-569.

[6] GAMA J, RODRIGUES P P. An overview on mining data streams[J]. Foundations of Computational, 2009, 6: 29-45.

[7] SUN R, ZHANG S, YIN C, et al. Strategies for data stream mining method applied in anomaly detection[J]. Cluster Computing, 2019, 22(2): 399-408.

[8] SILVA J A, FARIA E R, BARROS R C, et al. Data stream clustering: A survey[J]. ACM Computing Surveys (CSUR), 2013, 46(1): 13.

[9] LI Y, YANG G, HE H, et al. A study of large-scale data clustering based on fuzzy clustering[J]. Soft Computing, 2016, 20(8): 3231-3242.

[10] ZHANG P, SHEN Q. Fuzzy c-means based coincidental link filtering in support of inferring social networks from spatiotemporal data streams[J]. Soft Computing, 2018, 22(21): 7015-7025.

[11] AGGARWAL C C, HAN J, WANG J, et al. A framework for projected clustering of high dimensional data streams[C]// Proceedings of the Thirtieth International Conference on Very Large Databases. Netherlands: Elsevier Science Ltd, 2004: 852-863.

[12] CARNEIN M, ASSENMACHER D, TRAUTMANN H. An empirical comparison of stream clustering algorithms[C]// Proceedings of the Computing Frontiers Conference. New York: ACM Press, 2017: 361-366.

[13] DUBEY A K, GUPTA R, MISHRA S. Data stream clustering for big data sets: A comparative analysis[C]//Proceedings of IOP Conference Series: Materials Science and Engineering.[S.l.]: IOP Publishing, 2021, 1099(1): 012030.

[14] ZUBAROĞLU A, ATALAY V. Data stream clustering: A review[J]. Artificial Intelligence Review, 2021, 54(2): 1201-1236.

[15] CARNEIN M, TRAUTMANN H. Optimizing data stream representation:An extensive survey on stream clustering algorithms

[J]. Business & Information Systems Engineering, 2019, 61(3): 277-297.
[16] YIN C, ZHANG S, YIN Z, et al. Anomaly detection model based on data stream clustering[J]. Cluster Computing, 2019, 22(1): 1729-1738.
[17] HUANG L, WANG C D, CHAO H Y, et al. MVStream: Multiview data stream clustering[J]. IEEE Transactions on Neural Networks and Learning Systems, 2019, 31(9): 3482-3496.
[18] MANSALIS S, NTOUTSI E, PELEKIS N, et al. An evaluation of data stream clustering algorithms[J]. Statistical Analysis and Data Mining, 2018, 11(4): 167-187.
[19] GAMA J, PINTO C. Discretization from data streams: Applications to histograms and data mining[C]//Proceedings of the 2006 ACM Symposium on Applied Computing. [S.l.]: ACM, 2006: 662-667.
[20] RODRIGUES P P, GAMA J, PEDROSO J P. ODAC: Hierarchical clustering of time series data streams[C]//Proceedings of the 2006 SIAM International Conference on Data Mining. [S.l.]: Society for Industrial and Applied Mathematics, 2006: 499-503.
[21] DOMINGOS P, HULTEN G. Mining high-speed data streams[C]//Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. [S.l.]: ACM, 2000: 71-80.
[22] AGGARWAL C C. Data streams: An overview and scientific applications[J]. Scientific Data Mining and Knowledge Discovery, 2009: 377-397.
[23] GHESMOUNE M, LEBBAH M, AZZAG H. A new growing neural gas for clustering data streams[J]. Neural Networks, 2016, 78: 36-50.
[24] CAO F, ESTERT M, QIAN W, et al. Density-based clustering over an evolving data stream with noise[C]//Proceedings of the 2006 SIAM International Conference on Data Mining. [S.l.]: Society for Industrial and Applied Mathematics, 2006: 328-339.
[25] STREHL A, GHOSH J. Cluster ensembles—A knowledge reuse framework for combining multiple partitions[J]. Journal of Machine Learning Research, 2002, 3(12): 583-617.
[26] RAND W M. Objective criteria for the evaluation of clustering methods[J]. Journal of the American Statistical Association, 1971, 66(336): 846-850.
[27] O'CALLAGHAN L, MISHRA N, MEYERSON A, et al. Streaming-data algorithms for high-quality clustering[C]//Proceedings of 18th International Conference on Data Engineering. [S.l.]: IEEE, 2002: 685-694.
[28] PUSCHMANN D, BARNAGHI P, TAFAZOLLI R. Adaptive clustering for dynamic IoT data streams[J]. IEEE Internet of Things Journal, 2016, 4(1): 64-74.
[29] DE ANDRADE SILVA J, HRUSCHKA E R, GAMA J. An evolutionary algorithm for clustering data streams with a variable number of clusters[J]. Expert Systems with Applications, 2017, 67: 228-238.
[30] 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. Netherlands: Elsevier Science Ltd, 2003: 81-92.
[31] ZHOU A, CAO F, QIAN W, et al. Tracking clusters in evolving data streams over sliding windows[J]. Knowledge and Information Systems, 2008, 15(2): 181-214.
[32] UDOMMANETANAKIT K, RAKTHANMANON T, WAIYAMAI K. E-stream: Evolution-based technique for stream clustering[C]//Proceedings of International Conference on Advanced Data Mining and Applications. Berlin: Springer, 2007: 605-615.
[33] LÜHR S, LAZARESCU M. Incremental clustering of dynamic data streams using connectivity based representative points[J]. Data & Knowledge Engineering, 2009, 68(1): 1-27.
[34] FAHY C, YANG S, GONGORA M, et al. Ant colony stream clustering: A fast density clustering algorithm for dynamic data streams[J]. IEEE Transactions on Systems, Man, and Cybernetics, 2019, 49(6): 2215-2228.
[35] TASOULIS D K, ROSS G, ADAMS N M. Visualising the cluster structure of data streams[C]//Proceedings of International Symposium on Intelligent Data Analysis. Berlin: Springer, 2007: 81-92.
[36] KRIEGEL H P, KRÖGER P, NTOUTSI I, et al. Density based subspace clustering over dynamic data[C]//Proceedings of International Conference on Scientific and Statistical Database Management. Berlin: Springer, 2011: 387-404.

[37] NTOUTSI I, ZIMEK A, PALPANAS T, et al. Density-based projected clustering over high dimensional data streams[C]// Proceedings of the 2012 SIAM International Conference on Data Mining. [S.l.]: Society for Industrial and Applied Mathematics, 2012: 987-998.

[38] AMINI A, SABOOHI H, HERAWAN T, et al. Mudi-stream: A multi density clustering algorithm for evolving data stream [C]//Proceedings of Data Mining Workshops (ICDMW), 2013 IEEE 13th International Conference on. [S.l.]: IEEE, 2013.

[39] HYDE R, ANGELOV P, MACKENZIE A R. Fully online clustering of evolving data streams into arbitrarily shaped clusters [J]. Information Sciences, 2017, 382: 96-114.

[40] YIN C, XIA L, ZHANG S, et al. Improved clustering algorithm based on high-speed network data stream[J]. Soft Computing, 2018, 22(13): 4185-4195.

[41] HASSANI M, SPAUS P, CUZZOCREA A, et al. I-hastream: Density-based hierarchical clustering of big data streams and its application to big graph analytics tools[C]//Proceedings of 2016 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid). [S.l.]: IEEE, 2016: 656-665.

[42] CHEN Y, TU L. Density-based clustering for real-time stream data[C]//Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. [S.l.]: ACM, 2007: 133-142.

[43] WAN L, NG W K, DANG X H, et al. Density-based clustering of data streams at multiple resolutions[J]. ACM Transactions on Knowledge Discovery From Data (TKDD), 2009, 3(3): 14.

[44] PARK N H, LEE W S. Statistical grid-based clustering over data streams[J]. ACM Sigmod Record, 2004, 33(1): 32-37.

[45] DANG X H, LEE V, NG W K, et al. An EM-based algorithm for clustering data streams in sliding windows[C]//Proceedings of International Conference on Database Systems for Advanced Applications. Berlin: Springer, 2009: 230-235.

[46] SMITH T, ALAHAKOON D. Growing self-organizing map for online continuous clustering[C]//Proceedings of Foundations of Computational Intelligence. Berlin: Springer, 2009: 49-83.

[47] WANG C D, LAI J H, HUANG D, et al. SVStream: A support vector-based algorithm for clustering data streams[J]. IEEE Transactions on Knowledge and Data Engineering, 2011, 25(6): 1410-1424.

[48] ZHU Y, CHEN S. Growing neural gas with random projection method for high-dimensional data stream clustering[J]. Soft Computing, 2020, 24(13): 9789-9807.

[49] GUHA S, MEYERSON A, MISHRA N, et al. Clustering data streams: Theory and practice[J]. Knowledge & Data Engineering IEEE Transactions on, 2003, 15(3): 515-528.

[50] BIFET A, HOLMES G, PFAHRINGER B, et al. MOA: Massive online analysis, a framework for stream classification and clustering[C]//Proceedings of the First Workshop on Applications of Pattern Analysis. [S.l.]: PMLR, 2010: 44-50.

[51] HAHSLER M, BOLANOS M, FORREST J. Introduction to stream: An extensible framework for data stream clustering research with R[J]. Journal of Statistical Software, 2017, 76(1): 1-50.

[52] HAHSLER M, BOLANOS M, FORREST J. streamMOA: Interface for MOA stream clustering algorithms[J]. R Package Version, 2015(1): 1-2.

[53] HASSANI M, KIM Y, SEIDL T. Subspace MOA: Subspace stream clustering evaluation using the MOA framework[C]// Proceedings of International Conference on Database Systems for Advanced Applications. Berlin: Springer, 2013: 446-449.

[54] BIFET A, MANIU S, QIAN J, et al. Streamdm: Advanced data mining in spark streaming[C]//Proceedings of 2015 IEEE International Conference on Data Mining Workshop (ICDMW). [S.l.]: IEEE, 2015: 1608-1611.

作者简介:

朱颖雯(1982-),**通信作者**,女,副教授,研究方向:机器学习、人工智能、数据挖掘等,E-mail:yingwen.zhu@nuaa.edu.cn。

陈松灿(1962-),男,教授,研究方向:模式识别、机器学习、神经计算等,E-mail:s.chen@nuaa.edu.cn。

(编辑:刘彦东)