第53卷第5期 南京航空航天大学学报 Vol. 53 No. 5
2021年10月 Journal of Nanjing University of Aeronautics & Astronautics Oct. 2021

DOI:10.16356/j.1005-2615.2021.05.002

面向大数据的数据处理与分析算法综述

周 宇,曹英楠,王永超

(南京航空航天大学计算机科学与技术学院/人工智能学院,南京 211106)

摘要: 大数据处理是近年来广受关注和研究的技术领域,数据挖掘作为从大量数据中挖掘隐藏价值信息的技术,是处理大数据的有效工具。本文主要从数据挖掘的角度对大数据处理算法的研究现状进行分类总结。首先介绍了大数据中针对流式数据分类的方法,包括单模型算法和集成分类算法;其次分别从单机算法和基于分布式并行平台的多机算法两个角度概括介绍了大数据聚类方法以及大数据关联规则挖掘方法;最后总结了现有面向大数据的数据挖掘算法的研究进展并展望未来的发展趋势。

关键词: 大数据分类算法;大数据聚类算法;大数据关联规则挖掘

中图分类号: TP391.41 **文献标志码:** A **文章编号:** 1005-2615(2021)05-0664-13

Overview of Data Processing and Analysis Algorithms for Big Data

ZHOU Yu, CAO Yingnan, WANG Yongchao

(College of Computer Science and Technology/College of Artificial Intelligence, Nanjing University of Aeronautics & Astronautics, Nanjing 211106, China)

Abstract: Big data processing is a technical field that has received wide attention and research in recent years. Data mining, as a technology for mining hidden valuable information from a large amount of data, is an effective tool for processing big data. This paper mainly classifies and summarizes the research status of big data processing algorithms from the perspective of data mining. Firstly, the methods of big data classification for streaming data are introduced, including single-model algorithms and integrated classification algorithms. Secondly, the clustering methods and association rule mining methods for big data are summarized respectively from the perspective of single-machine algorithms and multi-machine algorithms based on distributed parallel platforms. Finally, the existing research progress of the big data-oriented data mining algorithm is summarized and the prospect of future development trend is put forward.

Key words: big data classification algorithms; big data clustering algorithms; big data association rule mining

随着多核CPU性能、存储设备性价比的迅速提升、网络带宽的不断增加,以及移动互联网、物联网、云计算等新一代信息技术的迅速发展和快速普及,人类生产生活已和信息技术相互交融,全球数据量出现爆发式增长。海量集聚的数据中蕴含着前所未有的社会价值和商业价值,政府和企业可通过分析处理大规模的用户数据得到传统数据分析方式无法获知的隐藏价值和模式,从而为公众和客户提供更加迅速、准确的公共服务及产品,大数据研究领域也因此吸引了社会各界的广泛关注。

大数据[1]指的是利用传统的数据处理分析技术或软硬件工具无法在可容忍的时间范围内对数据进行采集、处理和分析的数据集。其相较于传统的数据集,拥有4V特点[1]:规模性(Volume)、多样性(Variety)、价值(Value)和实效性(Velocity)。

基金项目: 国家自然科学基金(61972197)资助项目;江苏省自然科学基金(BK20201292)资助项目;青蓝工程资助项目。

收稿日期: 2021-06-21;**修订日期:** 2021-08-02

通信作者: 周宇,男,教授,E-mail:zhouyu@nuaa.edu.cn。

引用格式: 周宇,曹英楠,王永超. 面向大数据的数据处理与分析算法综述[J]. 南京航空航天大学学报,2021,53(5):664-676. ZHOU Yu, CAO Yingnan, WANG Yongchao. Overview of data processing and analysis algorithms for big data[J]. Journal of Nanjing University of Aeronautics & Astronautics, 2021, 53(5):664-676.

根据挖掘任务的不同,数据挖掘处理方法可分为分类分析、聚类分析、关联规则挖掘和序列分析等多种不同的方法。Blackett将各处理方法进一步地按数据分析的程度划分为3类[2]:使用历史数据进行总结的描述性分析、利用相关技术预测未来概率或趋势的预测性分析、发现隐藏数据相关性以帮助用户制定决策的规则性分析。聚类分析被用于描述数据,其通过衡量数据之间的相似性对数据进行分类;分类分析根据训练数据获得分类规则,故而常被用于预测分析;关联规则挖掘目的在于从数据库中发现并提取有价值的隐藏模式,且其思路同样适用于序列模式发现。其中,分类属于监督学习,聚类和关联规则挖掘属于无监督学习,这3类方法作为常用机器学习算法,在学术界受到较多关注和研究,发展出了许多方法和理论,本文将按照分类、聚类和关联规则挖掘依次对相关内容进行展开。

大数据的4V特征及其影响意味着数据的高复杂度和高计算成本,故难以使用传统数据挖掘方法来分析处理数据。在分类问题中,传统的对静态数据处理的方法已比较完善,但流式数据作为大数据的一个重要来源,传统算法难以对流式数据处理中出现的概念漂移现象进行应对,故针对大数据中流式数据的分类算法比较关键。聚类和关联规则挖掘方法最初便用于从各种应用的数据库中发现隐藏的有价值信息,因此在大数据时代,相关算法也受到越来越多的关注,面对大数据处理的挑战,关键在于如何通过并行传统算法或是提出新的算法来应对。

1　面向大数据的流式数据分类算法

本文按照图1的思维框架图,首先介绍流式数据的分类算法,分类作为一种监督学习方法[3],其目的是通过训练数据集中的实例以及实例所属的类标签发现分类规则,并使用该分类规则来预测未知实例所属类标签。分类过程有两个阶段:(1)分类规则学习阶段,该阶段通过分析、归纳训练集中各实例类标签情况,发现分类规则,创建合适的分类器模型;(2)测试集的分类阶段,该阶段需在验证集上评估分类器模型的分类准确率,若准确度较好,则使用该分类器对测试集中实例进行预先分类,否则需重新训练分类器模型。

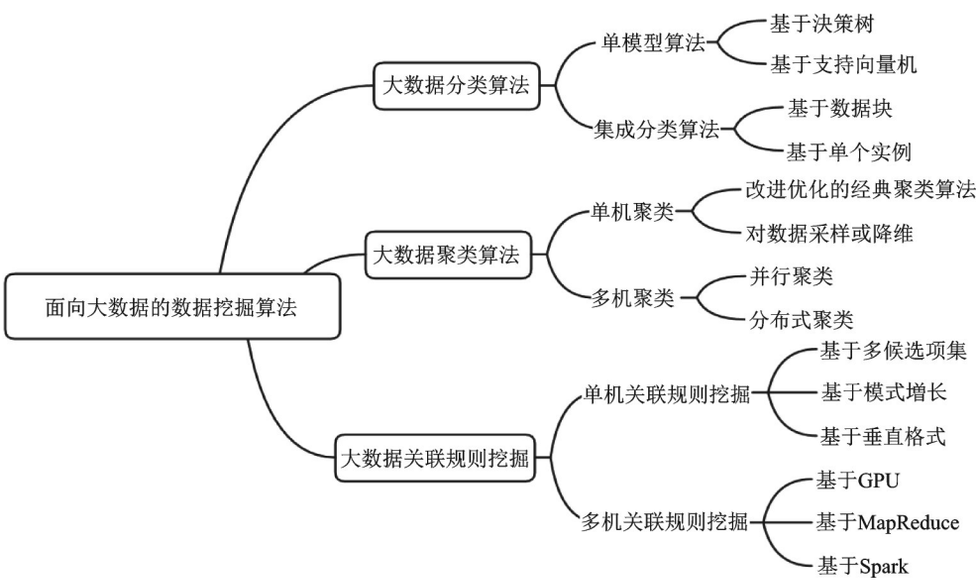

图1　面向大数据的数据挖掘算法划分

Fig.1　Taxonomy of data mining techniques for big data

1.1　流式数据特点

在大数据时代,流式数据作为大数据的一个重要来源[4-5],大多源自日常生活中的简单操作,例如网上购物数据、网络社交信息以及web应用程序生成的日志信息等。其具有如下主要特性[6]:

(1)无限性。流式数据作为真实世界中的记录,数据量是连续不断、无限的。

(2)实时性。各类数据源产生的数据是实时的,不可预知的。

(3)时序性。数据按产生的时间先后顺序按序到达。

(4)一次处理。因流式数据是实时且数据量无限,故流式数据中的数据通常只处理一次,在没有特别保存的情况下,流式数据不能被再次处理。

(5)富含变化。流式数据随时间不断产生的过程中,其数据分布动态改变,即会出现概念漂移的现象[7]。

1.2　概念漂移及其处理技术

概念漂移指流式数据中数据分布随时间发生

变化,导致实例和实例所属类标签之间的分类规则发生改变。如图2所示,常见的概念漂移有如下几种类型[8]:

(1) 突变漂移。概念在短时间发生巨大的变化且不可逆。

(2) 渐变漂移。概念随着时间逐渐的变化,且中间可能发生往复。

(3) 增量漂移。概念随着时间缓慢的不可逆的演变。

(4) 重复漂移。是一种临时性的改变,在短时间内会恢复成之前的状态。

(5) 罕见漂移。概念的异常改变。

(6) 噪声漂移。概念会由于数据中的噪声随机改变,但不是真正的概念漂移。

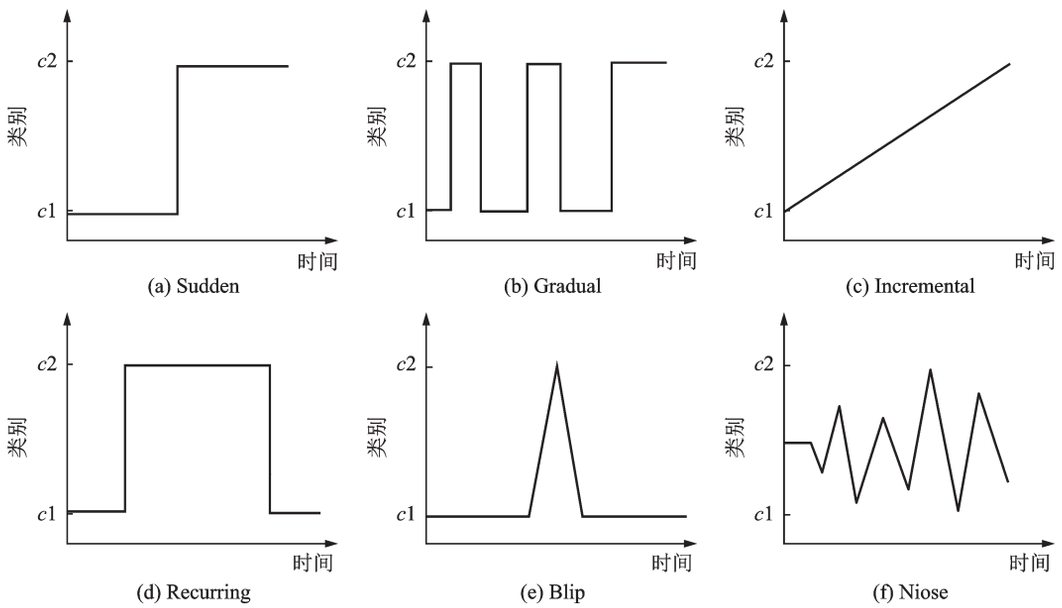

图2 概念漂移分布类型

Fig.2 Conceptual drift distribution type

流式数据分类关注的是实例和所属类标签之间最新的分类规则,故相关算法应具备一定的遗忘能力,使得所学习的分类器模型与最新的概念相一致。当概念漂移发生时,常用的几种概念漂移处理技术如下[7]:

(1) 滑动窗口技术。滑动窗口作为一个缓冲区,在缓冲区内的数据实例能反映当前最新的数据分布状况,且该窗口可以实时地对旧实例进行遗弃。当算法检测到概念漂移时,缩小窗口范围,否则适当地增加窗口大小,确保当前训练数据与最新数据分布保持一致,从而确保分类模型的准确性。

(2) 概念漂移检测技术。通过统计量比如误差、分错率等来监视数据分布的变化,是与给定分类器相合作的外部算法。如漂移检测法(Drift detection method,DDM)[9]使用统计量分类错误率和误差标准差,设置警告和漂移阈值,该算法一旦检测到数据分布开始变化并达到警告阈值,便向分类器模型发送警告信号,当达到漂移阈值时,表示概念漂移已经发生,需丢弃当前分类器重新学习,DDM算法能有效检测到突变漂移,但不擅长渐变漂移。基于Hoffding边界的在线漂移检测方法[10]对DDM算法进行了改进,其通过监控一系列性能指标的平均值,使用移动平均线和加权移动平均线,使得算法具备突变漂移和渐变漂移的检测能力。

(3) 集成学习技术[11]。将多个训练好的基分类器通过某种规则(投票、加权等)组合成一个新的分类器,从而得到比基分类器更好的分类效果。每个基分类器的训练数据不同,新的基分类器训练最近到达的数据,并淘汰掉性能最差的基分类器,使得集成分类器能适应数据分布的变化。

由于流式数据自身的5个特点,流式数据分类算法要求具备在资源受限的环境中实时地分析处理数据能力以及遇到概念漂移现象时具备快速恢复性和适应性能力。存在的流式数据分类算法按所含分类器个数主要分类两大类[11]:单模型算法和集成分类算法。

1.3 单模型算法

单模型算法,也称为单分类器方法,通过对传统批量分类算法的改进和扩展,并配备特定的概念漂移处理机制,使之能够适应流中数据的变化。根据传统分类算法,单模型算法可以分为基于决策树、支持向量机和贝叶斯模型等几种类型。由于分类问题基本都是线性不可分问题,决策树作

为经典的非线性分类器,已发展出许多基于决策树的流式数据分类算法,同时支持向量机作为一个线性分类器,可以通过利用核函数将非线性数据映射到高维空间使其转变为线性可分问题。两类方法各有优势,且呈现出平行发展的趋势,因此本文主要介绍基于决策树和基于支持向量机的两类单模型算法。

1.3.1　基于决策树的单模型算法

决策树算法的主要思想[12]:从包含待分类样本全集的根节点开始,递归地将根节点或者叶子节点替换为内部节点(决策节点)进行样本属性的测试来生长树,直到当前叶子节点没有属性可以划分或属于一个类时,算法完成对决策树的构造。

Domingos 等提出了快速决策树(Very fast decision tree,VFDT)[13],该算法在决策节点的属性划分上采用信息熵增益和基尼指数作为评判标准,仅扫描流式数据一次且不保存,能对大量数据流进行分类,但其没有考虑发生概念漂移的情况。针对 VFDT 的该缺点,Hulten 等[14] 提出了概念自适应的快速决策树(Concept adaptive VFDT,CVFDT),该算法在 VFDT 基础上,引入了处理概念漂移的滑动窗口技术,其通过当前窗口的流式数据建立临时子树,并在每次窗口滑动时对决策树进行优化更新。为了进一步解决不确定数据流中概念漂移问题,刘志军等[15] 提出了自适应快速决策树(Adaptive fast decision tree,AFDT),该算法将数据流中的特征属性分为不确定数值属性和不确定分类属性,并与概念漂移问题紧密联系,通过不断将不确定数值属性转化为不确定分类属性来构建决策树。极速决策树(Extremely fast decision tree,EFDT)[16]与 VFDT 类似,不同之处是在对决策节点进行属性划分时,使用 Hoeffding 边界来确定最佳属性划分带来的增益,且最佳属性划分具有一定的置信度,使得其所需时间比 VFDT 要长,但分类准确率有明显提高。Pecori 等[17]提出的流式模糊决策树(Streaming fuzzy decision tree,SFDT)将模糊逻辑和模糊集理论中的一些元素与 VFDT 相结合,利用均匀模糊划分对每个输入属性进行离散化,每个节点在分裂时以模糊信息增益作为度量指标选择最佳的输入属性,该算法相比于 VFDT 较为复杂,但其能高效处理模糊数据和噪声数据,有效提高分类的精度。

1.3.2　基于支持向量机的单模型算法

支持向量机的主要思想是[18]:针对线性不可分的情况,通过使用一个预先定义的核函数导出的非线性映射,将原始线性不可分的样本映射到较高维的线性可分的特征空间中,在新的特征空间采用线性算法对样本的非线性特征进行线性分析。

Wang 等[19]提出了一种用于大规模内核在线训练的预算随机梯度下降算法(Budgeted stochastic gradient descent,BSGD),该算法通过迭代地接收标记的流式数据,利用随机梯度下降(Stochastic gradient descent,SGD)在相应的目标函数上更新模型权重以及动态更新支持向量的数目来解决概念漂移问题,当支持向量数目超出预定于上界时,使用删除、合并以及投影 3 种主要的预算维护策略来控制支持向量的数量。该算法能有效地处理大规模流式数据分类,而且拥有较高的分类准确率。Le 等[20]提出了用于大规模在线学习的近似向量机算法。算法首先将整个输入空间域用足够小的单元重叠分区覆盖,并构造出相应的覆盖核心集,当有新的流式数据实例到来时,找到包含该实例的单元并用相应的核心点来近似这个实例,最后在近似表示该实例的核心点以及实例本身中通过伯努利随机采样选择一个加入支持向量集中,实现对模型的增量更新。该算法利用当前数据不断增量更新模型,从而能灵活地处理概念漂移问题,并增加分类的准确性。

上述基于决策树的分类模型,易构建且可解释性强,可以结合概念漂移检测技术处理应对概念漂移问题。由于现实数据的高维性及不平衡特性,未来可以在决策节点分裂时的属性划分上做进一步研究,以提高分类的准确性。基于向量机的分类模型能学习特征空间的维数,利用在线学习技术增量的更新模型来解决概念漂移问题,但随着数据量的增加,其支持向量的数目也会不断增长,进而导致模型的可扩展性差,未来可以在动态更新支持向量数目、核心集覆盖等问题上进行研究,从而提高模型的准确性及扩展性。表 1 概述了基于决策树和基于 SVM 的单模型算法优缺点。

表 1　基于决策树和基于 SVM 的单模型算法优缺点

Table 1　Merits and demerits of single model algorithms based on DT and SVM

模型	优点	缺点
基于决策树	易理解;易生成;结合概念漂移检测技术处理应对概念漂移问题	训练时间长;易过拟合
基于支持向量机	能学习特征空间的维数;结合在线学习技术增量更新模型应对概念漂移问题	内核选择难;参数调整复杂

1.4　集成分类算法

集成学习算法,也称为多模型算法,根据不同时间段的数据训练 N 个基分类器,通过某种组合方式将各分类模型组合成集成分类器,并将各基分类器的结果按某种机制进行综合得到集成模型的

结果,使得集成模型的鲁棒性更强,并能够灵活地处理概念漂移问题。根据模型每次处理数据的数据量,可分为基于数据块的集成分类算法和基于单个实例的在线集成分类算法[21]。

1.4.1 基于数据块的集成分类算法

基于数据块集成的主要思想是[22]:数据以块的形式进行传输,每个块包含固定数目的训练实例,根据训练生成N个基分类器,淘汰准确率低的分类模型,在线更新集成模型,最后综合各分类器的分类结果,达到快速适应数据分布变化的目的。

文献[23]中提出了一种准确率更新集成算法(Accuracy updated ensemble,AUE),该算法采用大小为n的窗口和增量的基学习器,使得集成模型中基分类器都可根据当前块的训练实例增量式更新,从而能更快速地适应突变漂移。文献[24]提出的CVFDT更新集成算法(CVFDT update ensemble,CUE)使用VFDT训练基分类器,用最新数据块更新准确率低于随机猜测准确率的基分类器,并对训练数据采用bagging操作增加基分类器间相异度,其分类准确率和概念漂移适应度都较高。基于信息熵的集成分类算法(Ensemble classification algorithm based on information entropy,ECBE)[25]根据分类前后熵值差异检测概念漂移并决定各基分类器的权值,通过相邻概念漂移周期内各基分类器权值信息得出概念稳定时的权值下界,移除权值低于该下界的基分类器,既提高了分类准确率又能快速适应新概念。Sarnovsky等[26]提出了多样化动态类加权集成算法。该算法维护可变数量的异构的基分类器,根据基分类器的性能动态调整其权重,一旦权重低于给定阈值,即将该分类器从集成模型中移除。通过多样性调整投票的组合规则以及各基分类器预测的聚合最终获得集成模型结果,若结果不正确,则向集成模型中添加新的随机基分类器。由于基分类器的自适应机制和多样化,使得该集成模型能有效处理各类型的概念漂移。

1.4.2 基于单个实例的在线集成分类算法

基于单个实例的在线集成主要思想是[27]:数据流中的实例单独到达,一次仅处理一个训练实例并对其进行增量学习,不断地修正集成分类模型,从而可以在有限的时间和内存下对数据流分类做出快速反应。

Zhai等[28]提出的EBPegasos集成算法以在线核SVM作为基分类器,检测到概念漂移时,首先判断漂移对各基分类器的影响,若影响较大,则需移除性能最差的基分类器,并创建新的基分类器,若影响不大,则对各基分类器进行增量更新,使得集成模型在遗忘和利用旧概念之间达成良好的平衡。Shan等[29]提出的在线主动学习集成模型(Online active learning ensemble,OALEnsemble)由一个稳定分类器和一定数目的动态分类器构成,稳定分类器根据数据流分布变化的长期趋势,捕捉渐变漂移,动态分类器根据数据流最新的变化,及时更新基分类器,捕捉突变漂移,使得该算法对概念漂移有较强的处理应对能力。Canzian等[30]提出了一种感知器加权多数投票算法,该算法采用一组分布式的学习器,每个学习器可以采用不同类型的局部分类器生成局部预测,收集并使用加权多数投票规则聚合其余学习器的局部预测以生成最终预测,最后根据学习器的预测结果,通过感知器学习规则来动态更新其聚合权值,使其能灵活应该概念漂移问题。该算法能对分布式、异构以及动态的数据进行分类且性能较高。

集成学习和在线集成学习都是通过聚合策略将多个学习器进行聚合从而得到集成结果,通过在学习过程中结合概念漂移检测技术监测流式数据分布的变化来快速适应各类型的概念漂移,但这使得算法性能在一定程度上受到漂移检测技术的限制。为了进一步提高和改善集成模型的准确率和性能,未来可以向基分类器的多样化和高鲁棒性的漂移检测算法发展研究。

2 大数据聚类

大数据聚类是一种无监督学习方法[31],即不需要预先定义类别和训练样本,其目的是将未知的数据集分成若干簇,使得同一簇中的数据点具有尽可能大的相似度,而不同簇中的数据点具有尽可能大的相异性。聚类过程主要有两个阶段:(1)定义相似函数,将其作为判断数据之间是否相似的依据;(2)选择合适的聚类算法将数据对象划分到不同的簇中。大数据聚类分析方法可分为单机聚类方法以及多机器集群聚类方法[32]。单机聚类方法的实现主要是通过对经典聚类算法进行优化、扩展以及对数据集进行降维或采样等处理,而主流的多机聚类方法则主要体现在体系结构等方面,其底层聚类算法与经典聚类算法无本质差别。

2.1 单机聚类

为解决处理大规模数据聚类的需求,单机聚类算法也一直在演进,现在的单机聚类主要有2个研究方向:通过改进、优化经典的聚类算法,使算法本身即具备精确的聚类效果又有良好的可扩展性,或是在不改变数据分布特征的前提下,直接对待处理的数据集进行采样或降维等处理,即通过减少数据

量或缩小整个数据集的形式来处理大规模数据。

在经典聚类算法基础上进行改进、优化的可扩展单机聚类算法,如 Monath 等[33]提出的子聚类组件算法(Sub-cluster component algorithm, SCC),是一种凝聚型层次聚类,算法使用链接函数来衡量两组点的相似性,以最佳优先方式进行迭代,即迭代从最易决定相似性高低的簇开始,并延长相似度衡量困难的决定,直至其易判断,且算法的每次迭代都会以不同的粒度生成簇。该算法能生成高质量的聚类结果,且可以扩展到数十亿数据点而不牺牲质量。Kobren 等[34]提出了一种纯度增强旋转的层级聚类算法(Purity enhancing rotations for cluster hierarchies, PERCH),算法将新数据点路由到叶子上,以增量的方式进行树结构的构建,通过树的旋转纠正贪婪增量聚类过程中的错误,提高了子树的纯度、聚类的质量以及算法执行速度,并引入促进平衡的旋转,使得算法可扩展到大量数据点 N 和集群 K 的极端聚类上。

对待处理数据集进行采样或降维等处理的单机聚类算法,如 Zhao 等[35]提出了分层抽样加扩展的模糊 C-means 聚类算法。该算法首先通过局部敏感哈希技术将相似数据划分到一起,粗略地将大规模数据划分成若干层,并从各层中随机抽取数据对象形成代表性样本;其次在采样的样本上使用模糊 C-means 聚类算法生成局部聚类结果;最后使用最近邻近标记实现样本外扩展,即将样本外对象分配到距它们最近的、已确定的簇中。该算法通过对数据进行分层采样,保持了原始大规模数据集中的数据分布特征,提高了聚类质量和计算效率。Chen 等[36]提出了一种基于深度学习的深度流形聚类方法(Deep multi-manifold clustering, DMC),由于位于流形局部的点具有相似的表示,算法首先通过迭代最小化局部保持目标函数使得学习到的表示有意义,其次通过直接对表示执行 K-means 聚类算法确定聚类中心,并使用面向聚类的目标函数,即根据表示到各质心的接近程度来惩罚表示,来指导模型提取特定于聚类的表示,最后将两个目标函数集合为一个联合损失函数,使用反向传播不断地对其优化,使得算法具有更精确的聚类效果以及更强大的性能。Mautz 等[37]提出了一种在高维空间中具有轻量级参数的非冗余 K-means 聚类算法(Non-redundant K-means, Nr-K-means),该算法寻找数据集中相互正交的子空间,确保发现的聚类代表数据的不同视图,并在所有子空间中优 K-means 的目标函数,从而识别不同子空间中的多个非冗余簇,即数据对象在不同子空间中属于不同簇,可进一步分析各个非冗余簇之间的关系。算法引入的噪音空间能过滤不能在所有子空间中表示良好的数据特性,使得算法具备同时降维的能力以及高效地执行速度。

2.2　多机聚类

多机聚类首先将待处理数据划分成若干块,分别加载到不同机器上执行分组聚类,其次综合分组聚类结果并根据分析结果自动改进聚类过程,最后再重新进行分组聚类,直到综合聚类结果符合一定的终止条件。多机聚类具有较高的处理速度和可扩展性,但增加了机器间的通信成本,因此优秀的多机聚类算法需在尽可能少的通信成本下,获得较好的聚类效果。多机聚类可以分为并行聚类和分布式聚类[38]。

2.2.1　并行聚类

并行聚类主要是通过纵向扩展平台实现单机聚类算法的并行,常见的纵向扩展平台如多核 CPU,图形处理单元(Graphics processing unit, GPU)等。

基于多核 CPU 的聚类算法,如 Hadian 等[39]提出了基于多核 CPU 的并行 K-means 算法,该算法有两个线程集组成:主线程和块聚类线程。主线程负责在读取数据集时按预定义的大小划分块,并使用队列概念将块发送给块聚类线程。块聚类线程接受块后使用 K-means 算法进行聚类,得到当前块的质心并存储入全局质心列表中。所有块聚类完成时,主线程加载全局质心列表,通过 K-means 算法对质心进行聚类,形成最终聚类,在 12 核机器上的评估结果显示,该算法在产生的聚类质量上与 K-means、K-means^{++}相同但运行速度快得多。Erdem 等[40]提出了一种多核具有噪声的基于密度的空间聚类算法(Multicore fast density-based spatial clustering of applications with noise, M-FDBSCAN),该算法首先将二维模糊数据对象集按核数进行划分,其次对每个子集并行使用 FDBSCAN 算法进行聚类,获得部分确定聚类区域,最后合并成最终聚类区域,该算法性能随着核数的增加呈线性增长,处理速度明显增加。

基于 GPU 的聚类算法,如 Melo 等[41]提出了基于 GPU 并行化的点排序识别聚类结构(Ordering points to identify the clustering structure, OPTICS)实现方式,OPTICS 并行化有两个阶段,即图形构造和执行 OPTICS 算法,其中执行 OPTICS 时间较短,重点在于图形构造的并行化。算法使用 3 个向量分别存储顶点、邻接列表节点以及边的权值。在构造成图时,并行地对这 3 个向量进行顶点度计算、邻接索引计算、邻接表组装和排序操作,实验结果表明,该方法不仅降低了 OPTICS 算法的复杂

度,而且执行速度比基于CPU的版本快得多。

前文的几种基于多核CPU以及基于GPU的聚类算法中,多核CPU适用于复杂程度高的算法,GPU适用于低复杂度、高计算量的算法。为进一步优化算法的性能,未来可以探索和设计将多核CPU与GPU并行计算相结合,同时利用两者优势的算法。

2.2.2 分布式聚类

分布式聚类使用分布式计算框架自动对数据集进行划分和分配,并在集群上实现聚类算法的并行,更适合处理大数据聚类,常见的分布式计算框架有MapReduce、Spark等。

Li等[42]提出的Multiplex K-means算法,使用MapReduce并行执行多个具有不同质心组的K-means进程,Map函数负责计算当前点与所有质心间的距离,Reduce函数负责质心的更新,每一次迭代根据聚类内总变异值对其进行评估,对聚类质量差的K-means过程进行剪枝,避免无用计算,对保留的高质量质心群采用启发式方法生成新的质心群,扩大聚类结果的搜索范围,避免陷入局部最优解,重复此过程获得最终聚类结果,该算法相比于串行的K-means算法有更好的效果,也适用于大数据集聚类。He等[43]提出了一种MR-DBSCAN算法,该算法根据空间邻近度对数据进行分区,并在各分区独立执行DBSCAN聚类算法,全局聚类结果由局部聚类结果聚合而成,该算法子程序完全并行且可扩展性良好。胡健等[44]提出的一种基于改进的果蝇算法(Improved fruit fly optimization algorithm, IFOA)的并行密度聚类算法(The density-based clustering algorithm by using IFOA based on MapReduce,MR-DBIFOA),使用K-维树和贪心算法对数据进行划分,使得各分区内数据对象数量相近,并通过基于自适应搜索策略和聚类判定函数IFOA对局部聚类的参数进行态寻优,提升局部聚类效果,并结合MapReduce实现局部聚类的并行计算,合并得到全局聚类结果。该算法在大数据集下的聚类效果、寻优能力和鲁棒性方面效果良好并具有较强的并行性能。

基于Spark的聚类算法,如王博文等[45]提出利用Spark设计的并行K-means算法,首先随机选取多个数据点作为质心,其次在一个弹性分布式数据集(Resilient distributed dataset,RDD)变换中,并行计算当前数据实例与各质心的距离,将其划分到距离最近质心的所属类簇,接着将各类簇的均值点更新为新质心,若相邻两次质心的误差小于给定阈值,则得到最终聚类结果,反之,再次计算距离进行迭代。该算法有不错的聚类效果和并行能力。Liu等[46]提出了一种基于Spark RDD模型的分布式密度峰值聚类算法。该算法首先预定义局部密度阈值P和密度较高点的距离阈值D,其次通过调用Spark的图形计算API(GraphX)生成图并计算截断距离,然后计算每个数据点的局部密度p和与密度高点的距离阈值d,最后选择$p>P$且$d>D$的数据点作为聚类中心,选择$p<P$且$d>D$的数据点作为隔离点,将剩余点分配到与其最近的聚类中心,从而完成聚类,该算法比使用MapReduce实现的密度峰值聚类算法快10倍,能更高效地处理大数据聚类。

基于分布式计算框架的算法能有效处理大数据聚类且扩展性好,但其消耗的软硬件资源较多,未来可以向着在消耗软硬件资较少的前提下实现高效、精确及可扩展的大数据聚类算法的方向进行研究。

3 大数据关联规则挖掘

大数据关联规则挖掘指从大量数据中发现隐藏的信息关联性,并通过分析该关联性帮助决策者制定正确的决策。关联规则的强度可用支持度和置信度来衡量[47]:若有规则$X\rightarrow Y$,则支持度为事务X和Y在事务数据库中出现的频率,用于判断该规则是否有利用价值;置信度为当事务X出现时事务Y出现的频率,用于判断该规则的正确度。关联规则挖掘主要有两个步骤:(1)从数据中找出支持度不小于最小支持度的规则,称为频繁模式或频繁项集;(2)从频繁模式中找出置信度不小于最小置信度阈值的强关联规则,其中重点在于频繁模式的挖掘。关联规则挖掘可分为基本的单机关联规则挖掘和基于分布式计算框架的多机并行关联规则挖掘[48]。

3.1 单机关联规则挖掘

单机挖掘的基本方法有3种:基于多候选产生的Apriori算法[49]、基于模式增长的频繁模式增长算法(Frequent-pattern growth, FP-growth)[50]和基于垂直格式的Eclat算法[51],其余的单机算法大都是在基本方法的基础上,通过改善存储结构快速高效地对候选集剪枝以及快速获取候选集支持度而提出来的。

Apriori算法依据两个先验性质,即非频繁项集的超集一定是非频繁项集以及频繁项集的子集必为频繁项集,对事务数据库进行一遍扫描,并根据MinSup剪枝得到频繁1项集,然后连接频繁1项集得到候选2项集,再次扫描数据库剪枝获得频

繁2项集,以此类推进行迭代,直到无候选项集产生为止,最后根据MinCon筛选频繁项集得到强关联规则。该算法的优点在于对每次迭代进行剪枝,在一定程度上减少了计算量,但每次剪枝需访问数据库且算法执行过程中产生了大量候选项集,其导致算法的执行效率低下。

FP-growth算法主要包含两个阶段:(1)构造FP-Tree,扫描一次数据库得到频繁1项集,再次扫描去除事务中非频繁项目并将剩余项目按支持度递减的顺序进行排列,初始根节点为空,将事务依次插入到FP-Tree中,每条事务在插入时,事务的第1项作为根节点的子节点插入,若已含该节点,则只需将该节点的节点计数加1,否则将该项新生为根节点的子节点插入,在插入事务的第2项时,将其作为已插入节点的子节点并按相同方法进行插入,直至该事务的所有项目插入完毕。(2)自底向上地对FP-Tree进行递归挖掘,从频繁1项集中支持度最低的项目作为初始后缀开始,在FP-Tree中查找该后缀的条件模式基并构造其条件FP-Tree,递归地挖掘该条件FP-Tree的频繁模式并将其与初始后缀连接实现模式增长。该算法相较于Apriori有两个优势,即不产生候选项集以及只访问数据库2次,其不足之处在于压缩原始数据的FP-Tree需全部装载入内存,当数据量大时,会导致计算能力受限。

Apriori和FP-growth算法都以水平数据格式{TID:itemset}挖掘频繁项集,而Eclat算法采用的是垂直数据格式{item:TIDSET}。在支持度计算上,水平数据格式需访问一次数据库得到而垂直数据格式中支持度即为该项目的TID集长度。Eclat算法通过遍历1次事务数据库,将数据格式转换为垂直格式,并通过支持度计算和剪枝得到频繁1项集,将频繁1项集求交集、剪枝得到频繁2项集,迭代此过程直至无候选项集产生。该算法与Apriori算法都是通过频繁k项集生成频繁$k+1$项集,但其只需访问数据库1次,加快了候选项集的产生,且优化了支持度的计算方式,3种基本方法的优缺点对比见表2。

表2　传统关联规则挖掘算法对比总结

Table 2　Comparison and summary of traditional association rule mining algorithms

算法名称	数据格式	优点	缺点
Apriori	水平	易实现; 适合稀疏数据集	产生庞大候选项集; 需多次访问数据库
FP-growth	水平	不产生候选项集; 只访问数据库2次	FP-Tree需载入内存
Ecalt	垂直	优化了支持度的计算; 只需访问数据库1次	数据集大时,算法效率低下

吴磊[48]提出的基于事务映射区间求交算法(Interval interaction and transaction mapping, IITM)在FP-Tree基础上,采用深度优先遍历为每个节点生成区间,获得压缩程度更强的频繁1项集区间列表,并使用哈希结构进行存储。IITM使用频繁1项集和频繁k项集来生成候选$k+1$项集,并使用由频繁k项集生成的布隆过滤器对候选项集进行高效地剪枝,在支持度计算上,由两个区间列表求交得到候选$k+1$项集的区间列表,从区间列表中即可获得支持度,并筛选出频繁$k+1$项集。该算法使用区间进一步压缩数据,引入布隆过滤器降低候选项集的数目,通过区间求交快速获取候选项集的支持度,使得算法具有较高的效率。

3.2　多机并行关联规则挖掘

多机并行关联规则挖掘指将数据进行划分并分配到各节点当中,每个节点运用单机上的挖掘方法独立地完成频繁模式的挖掘,最后将各节点结果进行汇总得到最终结果,适合于大数据上的关联规则挖掘。常见的多机并行挖掘方式按照大数据处理平台的不同大致可分为基于GPU的并行关联规则挖掘、基于MapReduce的并行关联规则挖掘以及基于Spark的并行关联规则挖掘。

3.2.1　基于GPU的并行关联规则挖掘

张旭[52]提出了基于GPU的并行关联规则挖掘算法,该算法利用统一计算框架(Compute unified device architecture,CUDA)编程模型,在CPU主机端完成逻辑复杂的候选项集生成,在GPU设备端并行计算逻辑简单但计算量大的候选项集的支持度。算法依此主要分为两个阶段:(1)在CPU中进行,在深度为k的前缀树中,找到前$k-1$项相同的两个频繁k项集,将它们的第k项按字典顺序进行连接,得到候选$k+1$项集,并将存储结构由前缀树转换为位表结构,以便在GPU中进行计算;(2)在GPU中进行,将两个频繁k项集的位表进行按位与运算得到候选$k+1$项集的位表,使用多个并行线程计算其位表中“1”的数目,得出候选项集的支持度并判断其是否为频繁项集。该算法通过GPU加速了支持度计算速率,且算法的性能随着测试数据

集规模的增长逐步扩大。

Chon等[53]提出了基于GPU的GMiner算法，该算法采用第1层遍历策略(Travesal from the first level, TEL)和中间层跳转策略(Hopping from the intermediate level, HIL)充分发挥GPU的计算能力。TEL将频繁1项集映射到事务数据块内并划分得到若干分区，将分区和事务数据块传送至GPU中，在GPU中通过按位与得到候选项集的部分支持度，并传送入主存进行汇总。TEL仅对频繁1项集进行大量计算来挖掘全部频繁项集，在挖掘较长的频繁项集时性能较低，而HIL策略通过使用更多GPU显存来提高此性能。该算法充分利用GPU的计算能力，且其性能随着GPU数量的增加呈线性增长，适用于大规模数据集的挖掘。

Djenouri等[54]提出的CGSS(Single scan on multiple cluster nodes equipped with GPUs)算法结合了集群和GPU的能力，将挖掘过程进一步加速。在CGSS中，集群采用主从模式且每台机器都配有GPU设备端，该算法主要分为5个过程：(1)简单分区。主节点将事务数据库等分成p个分区分配工作节点。(2)智能分区。根据工作节点GPU中线程块数量r，各分区通过迭代将相邻事务依次分配给不同的线程块，划分成r个子分区。(3)各线程块生成所有可能的候选项集，并存入线程块的本地哈希表中。(4)合并。每个GPU工作节点合并其线程块的本地哈希表创建GPU本地哈希表，主节点在GPU本地哈希表的基础上创建1个全局哈希表。(5)生成频繁项集。该算法利用分区策略解决了集群负载的不均衡以及减少了线程发散，在面向大数据的关联规则挖掘上具有更高的性能和并行能力。

3.2.2 基于MapReduce的并行关联规则挖掘

(1) 并行Apriori算法

Huang等[55]提出的基于MapReduce的Smart-Cache算法，由两个MapReduce作业组成：第1个是在Map函数中，降低MinSup得到局部支持度，根据局部支持度执行局部挖掘算法，将支持度在两者之间的项集存入Cache中，通过简单的线性回归分析出Cache容量呈非线性增长的转折点，即最佳局部支持度阈值，将根据局部支持度筛选得到的局部频繁项集以及支持度信息写入Hadoop分布式文件系统(Hadoop distributed filesystem, HDFS)的Cache文件中，Reduce将Map的结果进行汇总；第2个是Map任务在HDFS的Cache文件中查找支持度信息，Reduce任务通过合并各个候选项集的支持度得到全局频繁项集。该算法使用局部支持度提高了结果的准确性，通过存储第1个MapReduce产生的支持度信息，大大减少了第2阶段的计算量，提高了执行效率和算法性能。

Chon等[56]提出的BIGMiner(BIG data pattern miner)算法，包括预计算和挖掘频繁k项集2个阶段。预计算阶段根据预定义集合长度将频繁1项集划分成互不相交的集合，枚举出所有子集，生成垂直数据格式的事物块。挖掘阶段中Map任务根据频繁k项集和事务块生成候选$k+1$项集及局部支持度，Reduce任务合并局部支持度获取频繁$k+1$项集，并将其广播到所有机器，用于下次迭代时Map任务中。该算法在执行过程中不产生中间数据，且只需广播频繁k项集的网络开销，使得算法伸缩性好，在大规模数据上具有较高性能。

(2) 并行FP-growth算法

Xun等[57]提出了基于频繁完全树(Frequent items ultrametric tree, FUI-Tree)的FiDoop算法，FUI-Tree相较于FP-Tree最大的优势在于避免了递归遍历。FiDoop算法首先通过2次扫描事物数据库，将各条事务中非频繁项目丢弃，得到含有k个频繁项目的k项集；其次将k项集分解成2～(k—1)个项集，使用长度相同的项集构造FUI-Tree；最后通过FUI-Tree的叶节点计数便可获得所有频繁项集。该算法实现了自动并行化，能够在大型集群上并行地进行数据挖掘，进一步加快了数据挖掘速度，面对高维数据集时，Xun等[57]还提出了通过维度减少的方法来提高挖掘效率的FiDoop-HD算法。

Wang等[58]提出了一种基于多项目支持的频繁模式挖掘算法(Multiple item support frequent patterns, MISFP-growth)。该算法首先在数据预处理阶段引入预定义的多项目支持度；其次用1个MapReduce作业完成对各项目实际支持度的计算，得到多项支持的最小值，删去事务中低于最小多项支持的项目，并将剩余各项目按实际支持度大小降序排序；最后根据多项支持划分成子事务块，与FP-growth算法类似，通过构建MISFP-Tree的后缀条件模式基以及条件模式树来递归地挖掘频繁模式。实验结果证明，该算法在多项支持的数据集上实现了高效的挖掘，且通过并行计算减少了执行时间，适合于大数据分析应用。

(3) 并行Eclat算法

Gole等[59]提出的ClustBigFIM是一种将聚类和频繁模式挖掘混合的算法。算法利用MapReduce计算模型首先对待处理大型数据集进行K-means聚类生成多个簇；其次对每一个簇并行执行Apriori算法挖掘频繁k模式，使用生成的前缀树得到TID列表，并将其合并成一个全局TID列表，以便于将水平格式转换为垂直格式；最后通过对前缀

树执行Eclat算法挖掘频繁$k+1$项集。该算法使用K-means算法对数据进行预处理,并对簇进行频繁项集挖掘,从而使得算法的执行效率更加地高效,能更快地处理大数据集。

张春等[60]提出了一种基于MapReduce的MREclatK算法。该算法的思路比较简单,整个并行计算过程由一个MapReduce作业迭代完成。具体地,首先执行一次MapReduce作业得到频繁1项集,接着在每次迭代过程中在Map端将数据格式转换为垂直格式,将剪枝后的频繁1项集和频繁k项集求并集获得候选$k+1$项集,求交集获得相应的支持度,在Reduce端合并候选$k+1$项集得到全局支持度,通过比较MinSup筛选得到频繁$k+1$项集。该算法将传统Eclat算法向MapReduce计算模型上进行了迁移,使得该算法能处理海量数据。

3.2.3　基于Spark的并行关联规则挖掘

(1) 并行Apriori算法

Rathee等[61]提出的无需生成候选项集的算法R-Apriori包含3个步骤:第1步获取频繁1项集,Map函数完成flatMap()和map()的转换操作,Reduce函数调用reduceByKey()计算每个项目的支持度并修剪出频繁1项集,存入布隆过滤器中;第2步Map函数将事务中不包含在布隆过滤器的项目进行修剪,利用修剪后的事务快速生成所有可能的项集对,再使用Reduce函数来计算支持度并得到频繁2项集;第3步由频繁k项集迭代生成频繁$k+1$项集。该算法通过步骤2消除了候选项集的生成,降低了计算复杂度,有效地提高了效率和性能。

Rathee等[62]提出的动态关联规则挖掘算法Adaptive-Miner,改进了R-Apriori算法的频繁模式挖掘过程,该算法根据上一次迭代中生成的频繁项集数量分别计算R-Apriori方法和常规Apriori方法的时间消耗成本,动态选择当下最优的方法执行此次迭代,使得其具有更高的挖掘速度和性能。

(2) 并行FP-Growth算法

Shi等[63]提出了一种基于Spark的分布式FP-Growth算法(Distributed FP-growth algorithm based on Spark,DFPS)。该算法首先将全局频繁1项集广播到各节点;其次分配条件模式库,将频繁1项集中各项目按支持度降序排列,除首个项目外为剩余各项目分配一个分区存放条件模式基,并将事物中的项目分发到相应的分区;最后进行并行挖掘,各分区间数据相互独立,无需传递消息,独立地使用传统FP-growth算法来挖掘频繁项集。该算法在并行挖掘阶段消除了节点间的通信负载,且无多余的候选项目集生成,使得DFPS在大数据环境下的执行速度、并行能力以及可扩展性都得到了显著提升。

吴磊[48]提出了PIITM算法,该算法将IITM算法迁移到Spark平台上对大规模数据进行分布式并行挖掘。算法中各节点根据频繁1项集生成局部FP-Tree,并将生成的条件模式基根据后缀发送至相应的节点上,数据相互独立的各节点独立地执行IITM算法挖掘局部频繁项集,最后由Reduce作业汇总各结果并筛选出频繁项集。该算法充分考虑了节点间的负载均衡,且扩展性强,适应大规模集群。

(3) 并行Eclat算法

冯兴杰等[64]提出了一种基于Spark的Eclat算法SPEclat,算法首先以(Item,BitSet)的形式获取并存储频繁1项集,其次将具有相同前缀的频繁k项集划分到同一节点上,最后在各节点上将频繁k项集进行自连接得到候选$k+1$项集,并根据MinSup得到频繁$k+1$项集。该算法优化了支持度的计算,减少了候选项集的数量,在处理大规模数据方面具有较高的性能,但其没有对节点进行负载均衡处理。

Huang等[65]提出了一种优化的Eclat算法(Balanced parallel eclat,BPEclat),该算法在SPEclat算法基础上,使用范围划分思想平衡计算节点负载,根据Apriori算法的先验性质删除不满足MinSup的非频繁项集,对候选项集的生成采取了进一步优化。实验结果表明该算法通过减少候选项集的数量,提高了时间效率,在处理大数据时更为可靠,并具有较好的可扩展性。

上述基于GPU、基于MapReduce以及基于Spark的并行关联规则挖掘算法的主要思想是将经典算法迁移至分布式计算平台,但随着数据量的不断更新增加,将会产生海量的关联规则,故如何压缩关联规则数量以及如何利用已有的结果来增量地进行关联规则的挖掘,都值得在未来深入探索。

4　结　　论

本文从数据挖掘角度介绍了大数据处理算法,分别从流式大数据分类算法、大数据聚类算法和大数据关联规则挖掘3方面梳理总结了近年来国内外专家学者取得的进展。随着CUDA、MapReduce和Spark等分布式并行计算平台的普遍应用,面向大数据的数据挖掘算法的研究取得了巨大的进步,但针对大数据真实性、高维和高噪声等特点,仍需在如下方面开展进一步研究:

(1) 增量式更新算法。对于当前大多面向大

数据的数据挖掘算法都根据固定大小的数据集进行分析和评估算法性能,但在信息时代新数据无时无刻不在增加,要求算法能对结果进行动态更新,因此需进一步研究处理新旧数据变化的增量式更新算法。

(2)适用于高维数据的算法设计。目前大多面向大数据的数据挖掘算法都属于低维度算法,而大数据普遍具有超高维度、高稀疏特性,其意味着数据的高复杂性和高计算成本,因此设计适用于处理高维数据的高性能算法值得在未来深入探索。

(3)分布式并行算法的设计和实现。现有面向大数据的分布式并行算法大多是经典算法向分布式计算平台的迁移,并通过改进数据划分方法达到负载均衡和降低通信开销的目的。鉴于这一问题,未来可以考虑根据大数据的特点并结合分布式计算平台自身优势,设计并实现高性能、高并行性以及低复杂度的分布式并行算法。

参考文献:

[1] MANYIKA J, CHUI M, BROWN B, et al. Big data: The next frontier for innovation, competition, and productivity[M]. Washington: McKinsey Global Institute, 2011.

[2] BLACKETT G. Analytics network-or analytics[EB/OL]. (2013-12-09)[2019-03-07]. http://www.theorsociety.com/Pages/SpecialInterest/AnalyticsNetwork_analytics.aspx, 2013.

[3] HAQUE A, KHAN L, BARON M. Sand: Semi-supervised adaptive novel class detection and classification over data stream[C]//Proceedings of the AAAI Conference on Artificial Intelligence. USA: AAAI, 2016: 1652-1658.

[4] KREMPL G, ŽLIOBAITE I, BRZEZIŃSKI D, et al. Open challenges for data stream mining research[J]. ACM SIGKDD Explorations Newsletter, 2014, 16(1): 1-10.

[5] AGGARWAL C C. Data classification: Algorithms and application[M]. New York: CRC, 2014.

[6] MUTHUKRISHNAN S. Data streams: Algorithms and applications[M]. Hanover: Now Publishers Inc, 2005.

[7] ZHANG Peng, ZHU Xinghua, SHI Yong. Categorizing and mining concept drifting data streams[C]//Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. USA: ACM, 2008: 812-820.

[8] JIE LIU , AN Jinliu, FAN Dong, et al. Learning under concept drift: A review[J]. IEEE Transactions on Knowledge and Data Engineering, 2018, 31(12): 2346-2363.

[9] BARROS R S, CABRAL D R, GONÇALVES J R P M, et al. RDDM: Reactive drift detection method[J]. Expert Systems with Applications, 2017, 90: 344-355.

[10] FRIAS-BLANCO I, DEL CAMPO-ÁVILA J, RAMOS-JIMENEZ G, et al. Online and non-parametric drift detection methods based on Hoeffding's bounds[J]. IEEE Transactions on Knowledge and Data Engineering, 2014, 27(3): 810-823.

[11] 周志华.机器学习[M].北京:清华大学出版社, 2016.
ZHOU Zhihua. Machine learning[M]. Beijing: Tsinghua University Press, 2016.

[12] BIFET A, ZHANG J, FAN W, et al. Extremely fast decision tree mining for evolving data streams[C]//Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Canada: ACM, 2017: 1733-1742.

[13] DOMINGOS P, HULTEN G. Mining high-speed data streams[C]//Proceedings of the sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Boston: ACM, 2000: 71-80.

[14] HULTEN G, SPENCER L, DOMINGOS P. Mining time-changing data streams[C]//Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. USA: ACM, 2001: 97-106.

[15] 刘志军,张杰,许广义.基于自适应快速决策树的不确定数据流概念漂移分类算法[J].控制与决策, 2016, 31(9): 1609-1614.
LIU Zhijun, ZHANG Jie, XU Guangyi. Classifying algorithm for concept drifting of uncertain data streams based on adapting fast decision tree algorithm[J]. Control and Decision, 2016, 31(9): 1609-1614.

[16] MANAPRAGADA C, WEBB G I, SALEHI M. Extremely fast decision tree[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. USA: ACM, 2018: 1953-1962.

[17] PECORI R, DUCANGE P, MARCELLONI F. Incremental learning of fuzzy decision trees for streaming data classification[C]// Proceedings of the 11th Conference of the European Society for Fuzzy Logic and Technology.Atlantis: [s.n.], 2019: 748-755.

[18] SHALEV-SHWARTZ S, SINGER Y, SREBRO N, et al. Pegasos: Primal estimated sub-gradient solver for SVM[J]. Mathematical Programming, 2011, 127(1): 3-30.

[19] WANG Zhuang, CRAMMER K, VUCETIC S. Breaking the curse of kernelization: Budgeted stochastic gradient descent for large-scale SVM training[J]. The Journal of Machine Learning Research, 2012, 13(1): 3103-3131.

[20] LE T, NGUYEN T D, NGUYEN V, et al. Approximation vector machines for large-scale online learning[J]. The Journal of Machine Learning Research,

2017, 18(1): 3962-4016.

[21] KHAMASSI I, SAYED-MOUCHAWEH M, HAMMAMI M, et al. Discussion and review on evolving data streams and concept drift adapting[J]. Evolving systems, 2018, 9(1): 1-23.

[22] BRZEZINSKI D, STEFANOWSKI J. Reacting to different types of concept drift: The accuracy updated ensemble algorithm[J]. IEEE Transactions on Neural Networks and Learning Systems, 2013, 25(1): 81-94.

[23] BRZEZIŃSKI D, STEFANOWSKI J. Accuracy updated ensemble for data streams with concept drift[C]//Proceedings of International Conference on Hybrid Artificial Intelligence Systems. Berlin: Springer, 2011: 155-163.

[24] 马宪哲.基于集成分类器的数据流分类算法研究[D].沈阳:东北大学, 2012.
MA Xianzhe. Study on data streams classification algorithms baesd on ensemble classifier[D]. Shenyang: Northeastern University, 2012.

[25] WANG Junhong, XU Shuliang, DUAN Bingqian, et al. An ensemble classification algorithm based on information entropy for data streams[J]. Neural Processing Letters, 2019, 50(3): 2101-2117.

[26] SARNOVSKY M, KOLARIK M. Classification of the drifting data streams using heterogeneous diversified dynamic class-weighted ensemble[J]. Peer Computer Science, 2021, 7: e459.

[27] 翟婷婷,高阳,朱俊武.面向流数据分类的在线学习综述[J].软件学报, 2020, 31(4): 912-931.
ZHAI Tinging, GAO Yang, ZHU Junwu. Survey of online learning algorithms for streaming data classification[J]. Journal of Software, 2020,31(4): 912-931.

[28] ZHAI Tingting, GAO Yang, WANG Hao, et al. Classification of high-dimensional evolving data streams via a resource-efficient online ensemble[J]. Data Mining and Knowledge Discovery, 2017, 31(5): 1242-1265.

[29] SHAN Jicheng, ZHANG Hang, LIU Weike, et al. Online active learning ensemble framework for drifted data streams[J]. IEEE Transactions on Neural Networks and Learning Systmes, 2018, 30(2): 486-498.

[30] CANZIAN L, ZHANG Y, VAN DER SCHAAR M. Ensemble of distributed learners for online classification of dynamic data streams[J]. IEEE Transactions on Signal and Information Processing over Networks, 2015, 1(3): 180-194.

[31] AGGARWAL C C, REDDY C K. Data clusterin: Algorithms and applications[J]. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series, 2014, 31: 648.

[32] SHIRKHORSHIDI A S, AGHABOZORGI S, WAH T Y, et al. Big data clustering: A Review[C]//Proceedings of International Conference on Computational Science and Its Applications. Switzerland: Springer, 2014: 707-720.

[33] MONATH N, DUBEY K A, GURUGANESH G, et al. Scalable hierarchical agglomerative clustering[C]//Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. Singapore: ACM, 2021: 1245-1255.

[34] KOBREN A, MONATH N, KRISHNAMURTHY A, et al. A hierarchical algorithm for extreme clustering[C]//Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Canada: ACM,2017: 255-264.

[35] ZHAO Xingwang, LIANG Jiye, DANG Chuangyin. A stratified sampling based clustering algorithm for large-scale data[J]. Knowledge-Based Systems, 2019, 163: 416-428.

[36] CHEN Dongdong, LV Jiancheng, ZHANG Yi. Unsupervised multi-manifold clustering by learning deep representation[C]//Proceedings of Workshops at the Thirty-first AAAI Conference on Artificial Intelligence. San Francisco: AAAI, 2017: 385-391.

[37] MAUTZ D, YE W, PLANT C, et al. Discovering non-redundant K-means clusterings in optimal subspaces[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. Germany: ACM,2018: 1973-1982.

[38] DJOUZI K, BEGHDAD-BEY K. A review of clustering algorithms for big data[C]//Proceedings of 2019 International Conference on Networking and Advanced Systems (ICNAS). [S.l.]: IEEE, 2019: 1-6

[39] HADIAN A, SHAHRIVARI S. High performance parallel K-means clustering for disk-resident datasets on multi-core CPUs[J]. The Journal of Super-computing, 2014, 69(2): 845-863.

[40] ERDEM A, GÜNDEM T İ. M-FDBSCAN: A multicore density-based uncertain data clustering algorithm[J]. Turkish Journal of Electrical Engineering & Computer Sciences, 2014, 22(1): 143-154.

[41] MELO D, TOLEDO S, MOURÃO F, et al. Hierarchical density-based clustering based on GPU accelerated data indexing strategy[J]. Procedia Computer Science, 2016, 80: 951-961.

[42] LI Chen, ZHANG Yanfeng, JIAO Minghai, et al. Mux-Kmeans: Multiplex K-means for clustering large-scale data set[C]//Proceedings of the 5th ACM Workshop on Scientific Cloud Computing. Canada: ACM, 2014: 25-32.

[43] HE Yaobin, TAN Haoyu, LUO Wuman, et al. MR-DBSCAN: A scalable MapReduce-based DBSCAN algorithm for heavily skewed data[J]. Frontiers of Computer Science, 2014, 8(1): 83-99.

[44] 胡健,徐锴滨,毛伊敏.基于MapReduce和IFOA的并行密度聚类算法[EB/OL].(2020-04-03)[2021-05-13].https://doi.org/10.19734/j.issn.1001-3695.2020.

08.0187.
HU Jian, XU Kaibin, MAO Yimin. Density-based clustering algorithm by using improve fruit fly optimization based on MapReduce[EB/OL]. (2020-04-03)[2021-05-13]. https://doi.org/10.19734/j.issn.1001-3695.2020.08.0187.

[45] 王博文.基于K均值方法Spark化方法研究与实现[D].南京:南京财经大学,2018.
WANG Bowen. Parallelizing k-meas-based clustering on spark[D]. Nanjing: Nanjing University of Finance & Economics, 2018.

[46] LIU Rui, LI Xiaoge, DU Liping, et al. Parallel implementation of density peaks clustering algorithm based on spark[J]. Procedia Computer Science, 2017, 107: 442-447.

[47] TASSA T. Secure mining of association rules in horizontally distributed databases[J]. IEEE Transactions on Knowledge and Data Engineering, 2013, 26(4): 970-983.

[48] 吴磊.大数据环境下的频繁模式挖掘算法研究[D].广州:广东工业大学,2019.
WU Lei. Research on frequent pattern mining algotithm in big data environment[D].Guangzhou: Guangdong University of Technology, 2019.

[49] NEDUNCHEZHIAN R, GEETHANANDHINI K. Association rule mining on big data—A survey[J]. International Journal of Engineering Research, 2016, 5(5): 42-46.

[50] HAN Jiawei, PEI Jian, YIN Yiwem. Mining frequent patterns without candidate generation[J]. ACM Sigmod Record, 2000, 29(2): 1-12.

[51] HEATON J. Comparing dataset characteristics that favor the Apriori, Eclat or FP-Growth frequent itemset mining algorithms[C]//Proceedings of Southeast Con 2016. [S.l.]: IEEE, 2016: 1-7.

[52] 张旭.基于GPU的并行关联规则挖掘算法的设计与实现[D].北京:北京邮电大学,2015.
ZHANG Xun. Design and implementation of GPU-based association rules mining algorithm[D]. Beijing: Beijing University of Posts and Telecommunications, 2015.

[53] CHON K W, HWANG S H, KIM M S. GMiner: A fast GPU-based frequent itemset mining method for large-scale data[J]. Information Sciences, 2018, 439: 19-38.

[54] DJENOURI Y, DJENOURI D, BELHADI A, et al. Exploiting GPU and cluster parallelism in single scan frequent itemset mining[J]. Information Sciences, 2019, 496: 363-377.

[55] HUANG Dachuan, SONG Yang, ROUTRAY R, et al. Smart cache: An optimized mapreduce implementation of frequent itemset mining[C]//Proceedings of 2015 IEEE International Conference on Cloud Engineering(IC2E).USA: IEEE, 2015: 16-25.

[56] CHON K W, KIM M S. BIGMiner: A fast and scalable distributed frequent pattern miner for big data[J]. Cluster Computing, 2018, 21(3): 1507-1520.

[57] XUN Yaling, ZHANG Jifu, QIN Xiao. Fidoop: Parallel mining of frequent itemsets using mapreduce[J]. IEEE transactions on Systems, Man, and Cybernetics: Systems, 2015, 46(3): 313-325.

[58] WANG Chenshu, CHANG Juiyen. MISFP-growth: Hadoop-based frequent pattern mining with multiple item support[J]. Applied Sciences, 2019, 9(10): 2075.

[59] GOLE S, TIDKE B. ClustBIGFIM-frequent itemset mining of big data using pre-processing based on mapreduce framework[J]. International Journal in Foundations of Computer Science & Technology (IJFCST), 2015, 5(3): 79-89.

[60] 张春,汲磊举.基于MapReduce的Eclat改进算法研究与应用[J].北京交通大学学报,2016,40(3):1-6.
ZHANG Chun, JI Leiju. Research and application of Eclat improved algorithm based on MapReduce[J]. Journal of Beijing Jiaotong University, 2016, 40(3): 1-6.

[61] RATHEE S, KAUL M, KASHYAP A. R-A priori: An efficient apriori based algorithm on spark[C]//Proceedings of the 8th workshop on Ph d Workshop In information and Knowledge Management. Australia: ACM, 2015: 27-34.

[62] RATHEE S, KASHYAP A. Adaptive-Miner: An efficient distributed association rule mining algorithm on Spark[J]. Journal of Big Data, 2018, 5(1): 1-17.

[63] SHI Xiujin, CHEN Shaozong, YANG Hui. DFPS: Distributed FP-growth algorithm based on Spark[C]//Proceedings of 2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC). China: IEEE, 2017: 1725-1731.

[64] 冯兴杰,潘轩.基于Spark的并行Eclat算法[J].计算机应用研究,2019,36(1):18-21.
FENG Xingjie, PAN Xuan. Eclat algorithm based on Spark[J]. Application Research of Computers, 2019, 36(1): 18-21.

[65] HUANG Qiufeng, LI Qiang, HUANG Shiya, et al. Research on distributed parallel eclat optimization algorithm[C]//Proceedings of 2020 3rd International Conference on Artificial Intelligence and Big Data (ICAIBD). China: IEEE, 2020: 149-154.

(编辑:刘彦东)