基于分布式数据流的大数据分类模型和算法

毛国君 胡殿军 谢松燕
(中央财经大学信息学院 北京 100086)

摘 要 大数据是需求驱动的概念.随着数据库系统的普及和因特网服务的扩张,企业或者个人可用的数据正在膨胀,已有的技术很难满足大数据时代的数据分析需求,因此需要探索新的理论和方法来支撑大数据的应用.虽然大数据的 4V 属性已经被广泛讨论,但是它们大多描述的仍然是大数据的表象,所以很难从中抽象出统一的数据格式,因而进一步寻找可用于数据格式化的技术特征是必要的.面向于以分布式和流动性为主要技术特征的大数据应用需求,文中以分布式数据流为数据表达载体,在此基础上设计对应的大数据分类模型和挖掘算子.同时针对大数据的分类挖掘需要解决的关键问题来构建关键步骤对应的算法.理论上证明了文中给出的微簇合并技术和样本数据重构方法的合理性.实验表明:文中提出的基于分布式数据流的大数据的分类模型及算法不仅能大幅度地减少网络节点间的通讯代价,而且可以获得平均 10%左右的全局挖掘精度的提升(对比已有的典型算法 DS-means);虽然时间花费略高于 DS-means,但是两者在不同的数据容量测试下相差很小、且时间攀升趋势相当.

关键词 大数据;数据挖掘;分布式数据流;微簇;集成分类
中图法分类号 TP182 **DOI 号** 10.11897/SP.J.1016.2017.00161

Models and Algorithms for Classifying Big Data Based on Distributed Data Streams

MAO Guo-Jun HU Dian-Jun XIE Song-Yan
(*School of Information, Central University of Finance and Economics, Beijing* 100086)

Abstract Big data concept is from strong application requires. With the popularization of database systems and expansion of Internet services, the amount of data available to organizations and individuals is overwhelming, so current computing and analyzing techniques are suffering from efficient and effective bottlenecks. Therefore, it is necessary to explore related theory and technical methods to support the applications of big data. Although people have given the 4V attributes of big data, i.e. Volume, Velocity, Variability and Value, they are mostly presentational, and so it is still difficult to abstract the unified data expression form for various big data by directly using them. To solve this problem, we need look for more intensive technical features of big data. Aiming at the distributed and streaming feathers in technology, this paper first uses the concept of distributed data streams to express the big data, and then an efficient classifying model and its mining operators are presented. Finally, a series of algorithms based on this model are given to solve different problems in different phases for mining the big data with the distributed and streaming features. In theory, the rationalities of the key methods in this paper are proven. The experimental results show that our methods can achieve about 10% higher mining accuracies than typical methods such as DS-means doing, while the communication costs can be reduced in a large scale. Meanwhile, the mining time also can keep a reasonably climbing trend with increasing data sizes though it is a little more than DS-means.

Keywords big data; data mining; distributed data stream; micro-cluster; ensemble classifying

收稿日期:2015-09-30;在线出版日期:2016-04-05.本课题得到国家自然科学基金(62173293)和中央财经大学学科建设基金(CUFE00100101)资助.**毛国君**,男,1966 年生,博士,教授,中国计算机学会(CCF)会员,主要研究领域为大数据、数据挖掘及分布式计算.E-mail:maximmao@hotmail.com.**胡殿军**,男,1989 年生,硕士研究生,主要研究兴趣为大数据和数据挖掘.**谢松燕**,女,1992 年生,硕士研究生,主要研究兴趣为大数据和数据挖掘.

1 引 言

大数据(big data)概念最早是在 20 世纪 80 年代提出的,来自于《Nature》2008 年推出的大数据专刊,其强大的应用需求使之成为近年研究和应用的焦点.特别是,大数据潜在的应用前景已经被许多国家的政府关注,国家政府成为大数据技术的重要推动者.例如:2012 年,美国政府已经联合 6 个部门宣布了 2 亿美元的"大数据研究与发展计划";2015 年 9 月,中国国务院也印发了"促进大数据发展行动纲要".迄今为止,至少有 35 个国家出台了相关的政策或文件来支持大数据的发展.然而,虽然大数据的应用价值及前景已经得到国家层面、学者及商界的广泛认可,但是大数据作为新生事物,面对的技术问题仍然很多,许多的理论和方法也亟待解决,研究远远落后于应用的需求.

从研究角度看,目前出现的大数据概念可以大致归纳为如下几个流派:(1)数据论.强调大数据的核心是超大规模的数据集.典型观点来自于麦肯锡报告:"大数据指的是大小超出常规的数据库工具来获取、存储、管理和分析能力的数据集[1]";(2)方法论.强调大数据的核心是寻找新的分析方法.典型观点来自于维克托的《大数据时代》著作:"大数据指不用随机分析法(抽样调查)这样的捷径,而采用所有数据处理的方法[2]";(3)环境论.强调大数据必须结合相应的应用领域或者应用环境才有意义.如目前广泛讨论的社交大数据[3]、科学大数据[4]等;(4)特征论.强调大数据应该具有独特的属性.如国际数据公司(IDC)认为大数据有 4 个主要特征,即 4V 属性[5]:数据规模大(Volume)、数据聚集快(Velocity)、数据类型多(Variety)、数据价值大(Value).

这些研究从不同视角解释了大数据概念的内涵和外延.特别是 4V 属性集中概括了它的主要特征.然而,不容忽略的事实是目前大数据仍处于概念探索的初级研究阶段,特别是基础性研究远远落后于大数据概念的应用.我们通过中国知网,使用"大数据"作为关键词能检索出 30 多万篇文章,但是在限定计算机或者自动化相关期刊下,只有 6 万多篇.就是这 1/5 不到的文献绝大多数都是综述类的,专门涉及大数据挖掘的基础理论和方法的文献很少.这种"边研究边应用"的现状应该引起计算机及其相关学科研究人员的重视,当然也为其提供了广阔的研究空间.

事实上,大数据的 4V 属性给出的是大数据的表象特征,很难进行形式化研究.因此,结合应用特点找到更利于形式化抽象的技术特征可能是大数据技术研究的一个很好的突破口.例如,社交大数据一般是指来自于大型网站的网页数据,这样的数据不像数据库中数据表那样具有单一的数据结构,而是由结构化、半结构化及非结构化数据混合构成的,因此数据结构的多样性可能成为社交大数据的主要技术特征之一[2,5].再如,许多科学大数据具有很高维度的表达形式,表现出显著的高维性特征[4].类似地,图像、声音、视频等多媒体应用产生的大数据也是典型的高维数据.还有,在一个网络监控系统中收集的网络流量(network traffic)大数据,总是随着时间在不断增长的,而且数据隐藏的知识模式也是在动态流动中演化的,所以这类大数据就表现出典型的流式数据特征[6].类似地,电子商务网站采集的交易大数据也是典型的流式数据.这样,既然从大数据的 4V 属性出发很难进行数据的抽象表达,那么根据应用特点来寻找某类大数据的技术特征、进而进行形式化研究就是一条可行的技术路径.

以著名的电子商务网站 eBay 为例[1].它每天要处理 TB 级的数据流量,是典型的大规模数据.支持这种大规模数据处理的硬件环境则是几百台的数据服务器.硬件的分布性决定了数据收集是分布式的.此外,每台数据服务器上处理的数据都是按照交易时间来收集的,所以大数据的快速集聚属性对应着技术上的数据流动性.同时,如果我们只关心交易数据的话,那么这类大数据的数据结构是规范而简单的.所以,像 eBay 这样的电子商务网站上收集的大数据的主要技术特征就可以归纳为分布性和流动性,因而我们可以针对分布式和流动性为技术特征的大数据来进行相关的挖掘方法研究,且具有很好的理论和应用价值.

从概念演化角度上说,本文关注的这类大数据是目前已经出现的"流式大数据"[6]及"分布式大数据"[7]等概念的自然延伸.从应用价值上说,抽象出的分布式和流动性的技术特征在许多大数据的应用环境中存在.除了上面提到的电子商务网站的交易大数据外,分布式和流动性的大数据环境也可能发生在网络流量监控、传感器网络以及股票交易、银行业务、商品销售等交易系统中.

本文重点聚焦在具有典型的分布式和流动性技术特征的大数据的分类挖掘问题上.相比传统的分类挖掘技术,这类大数据的分类挖掘隐含着许多挑

战性的问题. 首先，传统的分类挖掘方法以单一学习样本集为基础，而大数据的分布式收集特性决定分类学习需要分布式进行，因而对应的分布式学习策略和方法需要研究；其次，动态流动的流式大数据和传统数据库存储的静态数据有显著的不同，不可能一次性将所有数据存储起来再进行离线式的挖掘，必须探索在线实时的收集技术和随时间变化的增量式的挖掘方法；最后，传统的分类挖掘技术对学习样本集要求较高，而分布式、流式大数据的分类挖掘需要多节点、多步骤协同处理，很难保证学习样本集的纯度，所以必须针对这类大数据的挖掘特点来探索鲁棒性能好的分类技术. 因此，面向于这类分布式的数据收集和随时间的数据流式聚集的大数据中的分类挖掘问题，需要集成化的技术和创新性的理论和方法来解决.

大数据的分类挖掘系统是一个多步骤协作工作的系统. 既有节点级的局部分析处理、又有全局性的模式发现，而且不同的阶段所要解决问题的侧重点也不同. 例如：面对潜在的快速流动、随时间不断集聚的流式大数据，局部节点处理应该更强调处理的实时性和高效性. 而全局的分类模式挖掘最主要的任务就是构造可以全局共享的分类器，所以更应该注重模式的预测能力和抗干扰性等. 因此，应该根据不同的挖掘阶段需求来研究对应的理论和方法，形成技术集成化、方法系统化的解决方案. 毋庸讳言，目前在大数据或者分布式数据流(distributed data stream)挖掘研究上还是以单一技术为主(如单独使用基于距离的分类模型[8]、基于频度挖掘的方法[9-10])，对于整体的解决方法研究还不够.

简言之，本文将重点关注具有分布式和流动性技术特征的大数据的分类挖掘问题，针对主要阶段的技术需求来探索有效的解决策略，通过研究对应的方法、并集成它们形成一个系统化的解决框架. 主要工作归纳为：借助于分布式数据流的数据形态来刻画这类以分布式和流动性为主要技术特征的大数据；在此基础上设计一个适合于这类大数据的分类挖掘模型；针对模型中的主要挖掘算子进行算法设计，并在理论和实验上评估这些算法的有效性.

2　相关工作

流式大数据的挖掘已经得到广泛关注，其中作为基础性的数据流挖掘(data stream mining)已经成为数据挖掘研究中的一个活跃分支. 数据流挖掘成果最多的是单数据流挖掘方法，其中增量式挖掘被广泛关注[11]，成为动态挖掘随时间流动的流式数据的有效方法. 滑动窗口(sliding window)被认为是解决数据流中动态知识发现问题的基本技术之一，它使得潜在无限数据流中的知识发现问题能够通过有限容量的数据窗口的叠加处理来解决[12-13]. 经过十余年的研究，(单)数据流挖掘已经积累了许多丰富的研究成果，它们构成了本文研究工作最基本的理论和技术基础.

为了高效地解决数据流挖掘问题，数据流的分布式挖掘研究出现. 2007 年，Parthasarathy 等人[14]给出了数据流的分布式挖掘需要面对的主要问题及对策，是较早地、比较全面地介绍数据流的分布式挖掘技术的文献之一. 他们的一个重要观点是：有效地利用有限的计算机处理资源来解决潜在无限数据的知识发现问题，必须寻找到一个代价与精度平衡的分布式(而不是集中式)解决方案. Bhaduri 等人[15]则从性能优化的角度说明了数据流的分布式挖掘需要综合考虑分布式计算、内存缓冲及节点交互代价等问题. 他们也设计了一个层次式的数据流的挖掘构架，即挖掘系统有若干局部节点和 1 个中心节点构成，通过局部节点的并行挖掘形成初级模式、中心节点再生成全局性的知识模式. 这些研究，特别是代价与精度平衡的挖掘思想，也是本文挖掘模型设计时需要重点考虑的问题.

当然，数据流的分布式挖掘关注的主要还是如何利用分布式并行技术来解决大容量的(单)数据流的知识发现问题. 事实上，随着基于网络的计算机应用系统的普及和壮大，多节点独立数据集聚但逻辑上关联的多数据流成为另一重要的数据形态，即分布式数据流. 诚然，目前关于分布式数据流挖掘的大多数文献还主要集中在关键科学问题及解决对策的研究上，但是随着科学问题越来越清晰，近年也出现了一些相关的挖掘构架及方法的讨论. 2012 年，Guerrieri 等人[8]提出了一个分布式数据流的挖掘算法 DS-means. 它的挖掘工作分成局部聚类、模式传输和全局聚类等 3 个关键步骤，是典型的层次式的分布式数据流挖掘构架. 在 DS-means 的局部及全局聚类中都使用了典型的无监督学习算法 k-means. 但是，为了适应不同节点、不同时间段上不同数据流的容量波动情况，在全局节点聚类之前先根据局部节点的聚类结果来动态地调整全局类簇的数目. 2014 年，Anceaume 等人[16]设计了多数据流的分布式数据评估算法 AnKLe，很好地解决了多节点并行

检测数据分布变化的问题. 2014 年,Cesario 等人[9]则面向于分布式数据流的频繁项目集挖掘问题构建了一个分布式多节点的挖掘构架,并设计了对应算法来实现该构架的关键操作. 本文工作借鉴了这些分布式数据流的挖掘构架和方法.

简言之,分布式数据流挖掘是数据挖掘及其数据流挖掘技术研究的自然延伸和发展,目前的研究主要以算法设计为主. 然而,面对日益增长的大数据需求,系统化地研究大数据的挖掘构架及其相关构架下的核心挖掘模型及算法成为一个必须面对的问题. 从这个意义上讲,分布式数据流可以为许多应用环境的大数据提供一种数据组织的理想模型,而分布式数据流的挖掘算法可以作为解决大数据挖掘的核心技术之一来使用.

事实上,随着大数据概念的日益升温,一些学者或者研究机构也开始尝试性探索大数据中的知识挖掘问题. 2012 年,Luo 等人[7]认为大数据的挖掘需要在不同处理阶段研究不同的算法,并且针对一些关键步骤设计了相应的算法. 2013 年,Wu 等人[17]从数据挖掘观点讨论了大数据分析问题,并且基于数据驱动策略探讨了大数据挖掘中的核心问题,并设计了一种大数据挖掘的体系构架和核心模型. 2014 年,孙大为等人[6]认为大数据有批量和流式处理两种数据形态,并对大数据流式计算中的关键技术进行了较为系统地分析.

大数据挖掘也是一个代价与精度的平衡优化问题,其中在合理的通讯代价下提升分布式挖掘的精度是关键的科学问题之一. 减少通讯代价的主要策略是节点间共享统计数据(而不是实时传输原始数据)[7,17]. 2008 年,Masud 等人[18]提出了一种在数据流中挖掘微簇(micro-cluster)模式的思想,即在对一个数据流执行聚类算法后抽取每个簇的点数、均值等统计值形成所谓的微簇模式. 2011 年,Kranen 等人则将微簇类似的数据统计信息设计成一种 ClusTree 的树结构,随着时间增长来维护它[19]. 以上两个工作都是面向于单数据流挖掘的. 一个直接面向于分布式数据流挖掘的相关方法来自于 Wang 等人[10]2011 年的工作. 它们设计了一个分布式数据流的频繁项目集挖掘框架,并通过维护数据概要(data synopsis)来实现节点间的信息交流,其中的数据概要模式和本文讨论的微簇模式的思想是一致的. 由于微簇挖掘生成的是数据的统计概要,所以对于分布式数据流分类挖掘来说,它可能是局部模式挖掘的理想解决方法之一. 这是因为传输微簇模式能有效地减少节点间的数据通讯代价(相比原始数据)、进而能有效地避免有限的网络带宽可能造成的通讯瓶颈问题. 微簇挖掘及其相关技术是本文局部模式挖掘方法的主要技术支撑.

虽然微簇挖掘在局部节点是有效的,但是由于微簇模式缺乏足够的预测能力,即对未知数据的分类能力,所以作为全局模式是不合适的. 事实上,在分布式数据流分类挖掘中,虽然数据是在局部节点分散收集的,但是分散在局部节点的数据流是相互关联的,所以发现全局分类器以用于多节点共享和预测是分布式、流式大数据的分类挖掘的主要任务之一. 此外,由于分布式挖掘要经过局部模式挖掘和数据传输才能进行全局模式挖掘,所以多步骤的处理可能导致数据质量的下降,因此全局模式的选择和设计也必须考虑抗噪性能. 基于以上两点,考虑到集成学习技术具有很高的预测能力和更好的鲁棒性能[20],所以本文将借鉴已有的集成学习技术,研究适合于分布式、流式大数据的分类挖掘需求的集成分类方法. 集成分类技术的研究相对比较成熟,可以借鉴的成果也较多[20-21].

3 大数据的分类模型

给定训练数据集 T 和类标识集合 C,分类学习就是从 T 中学习出一个分类器,而分类算法则是构建这种分类器的过程描述. 然而,流式大数据中的分类学习的训练数据集是随时间动态收集的,所以分类器的学习必然是一个动态的演化过程. 此外,传统机器学习的分类算法强调挑选出的训练集的质量,而且认为完整的分类学习需要通过正例集和负例集来学习. 即使是在数据流或者分布式数据流的相关研究中,也不乏关于基于抽样技术或者基于负例学习方法的研究[22]. 然而,在大数据的概念下,数据抽样和负例生成技术都不被推荐[2]. 事实上,面对庞大而快速流动的大数据,挑选高质量的训练样本数据集是不现实的,同时实时地构造出合适的负例样本集也是困难的.

大数据的挖掘首先要解决被分析数据的形式化表达问题. 如前所述,泛泛而谈的大数据概念隐含着形态各异的数据格式,很难统一地进行规范化描述. 然而,如果只关心这类具有统一逻辑视图的、多节点分布式采集的、随时间流动增长的数据形态,那么这类大数据可以借助于已有的(同构)分布式数据流的概念来完成数据的格式化抽象[8,15],形成可用于分

析的规范化数据形态.

定义 1(分布式数据流). 给定:时间序列 $T=\langle t_1,t_2,\cdots,t_i,\cdots\rangle$,数据维度 d 和节点数 n.一个分布式数据流被定义为:$S=\{S_1,S_2,\cdots,S_n\}$,其中每个 $S_k(k=1,2,\cdots,n)$是一个单数据流,是在 T 上采集的多维数据元组序列 $S_k=\langle r_1,r_2,\cdots,r_i,\cdots\rangle$,$r_i=(r_i^1,r_i^2,\cdots,r_i^d)$.

对于以分布式和流动性为主要技术特征的大数据的分类挖掘而言,定义 1 中界定的数据形态可以作为训练样本的收集模型来使用.例如,在诸如网络流量监测、电子商务交易等系统中,可以使用这样的数据模型来收集对应的训练数据.

事实上,随着收集时间点的增长,训练用的样本数据在不断地集聚,当然隐藏的知识模式也在发生变化.因此,流动性大数据的分类挖掘目标之一就是随着时间变化来及时更新分类器.然而,如果采用点到点的模式挖掘策略的话,即在每个收集点都进行分类器更新,虽然实时性很好,但是对于快速流动的流式大数据来说是不现实的.因此,本文将探索块到块的模式挖掘策略,即以数据块为单位来进行分类器的更新.这样,我们就需要改进传统的滑动窗口技术.定义 2 给出的历史窗口概念可以帮助解决这个问题.

定义 2(历史窗口). 给定:时间序列 $T=\langle t_1,t_2,\cdots,t_i,\cdots\rangle$ 和它上的一个数据流 $S=\langle r_1,r_2,\cdots,r_i,\cdots\rangle$.设 $t_i,t_k\in T$,$i<k$,则 $H_k=[t_i,t_k)$ 被称为 S 在 k 上的历史窗口,k 被称为一个挖掘点,同时在 H_k 内收集的数据 $chunk_k=\langle r_i,r_{i+1},\cdots,r_{k-1}\rangle$ 则被称为该历史窗口的数据块.对于一个分布式数据流来说,约定所有的节点都使用相同的挖掘点序列,而且在任意一个挖掘点 k 上,所有的局部节点都使用相同的历史窗口.

对每个局部节点来说,它们的主要任务就是挖掘局部模式,所以局部模式的表达是首先要解决的问题.面向于分布式、流式大数据的层次式挖掘构架[8-9],局部节点需要实时地维护自己的局部模式、并将它们传递到中心节点,因此数据传输量与局部模式的表达形式有直接的关系.当然是表达越紧凑越好.同时,由于局部模式最终要用于全局模式的学习,所以我们在选择局部模式表达形式时也需要考虑它的可恢复性,即利用它恢复全局学习样本数据的能力.基于这些考虑,本文在借鉴了已有的微簇(micro-cluster)挖掘研究成果的基础上,根据大数据的分布式和流动性的技术特点及其分类挖掘的需求,设计了定义 3 的微簇结构.与已有的相关研究相比[18],定义 3 通过增加平方和等统计变量来提高微簇的表达能力和微簇之间操作的方便性,同时也根据后面分类算法的需要增加了类标识项.

定义 3(微簇). 设 d 维的数据集 $X=\{x_1,x_2,\cdots,x_n\}$,其中 $x_i=(x_i^1,x_i^2,\cdots,x_i^d)$,则它对应的微簇结构由一个 5 元组 $M=\langle n,c,s,d,f\rangle$ 定义:

(1) $M.n$:数据个数(本例就是 n).

(2) $M.c$:中心点或称为均值,即

$$M.c^j=\Big(\sum_{i=1}^{n}x_i^j\Big)\Big/n,\ j=1,2,\cdots,d \tag{1}$$

(3) $M.s$:平方和统计(为防止溢出被开方),即

$$M.s^j=\sqrt{\sum_{i=1}^{n}(x_i^j\times x_i^j)},\ j=1,2,\cdots,d \tag{2}$$

(4) $M.d$:方差统计值,即

$$M.d^j=\Big(\sum_{i=1}^{n}(x_i^j-M.c^j)^2\Big)\Big/n,\ j=1,2,\cdots,d \tag{3}$$

(5) $M.f$:数据集的类标识.

定义 3 的微簇结构能够满足分布式、流式大数据的局部模式的紧凑型和可恢复性要求.这是因为一个微簇对应的是一个原始数据簇的统计信息,因而要比原始数据容量小得多,而且合适的统计信息要比更抽象的分类器(如 C4.5 等)更有利于全局学习数据样本点的恢复.

值得注意地是,微簇作为全局模式是不合适的.事实上,分布式、流式大数据的全局模式应该是对分布在多个节点的局部流动数据的共性归纳,而且主要任务是能够对流动的未知类别的数据进行分类预测,为分布式的多节点的流式数据提供共享的预测模型.很显然,微簇模式不具备这样的预测能力.因此,作为最终的知识模式,全局模式需要预测能力更强的知识模式形式.集成分类器具备这些优势,因此本文将研究对应的集成分类器方法来实现分布式流式大数据的全局模式挖掘.

简言之,面对复杂的分布式、流式大数据分类问题,本文将研究、改进、集成多种技术方法,其中包括上面提到的块到块的增量式更新策略、局部的微簇挖掘方法和全局的集成学习技术等.图 1 给出了一个从原始数据流到局部模式再到全局模式的增量式的挖掘过程示意.

在图 1 中,包含 3 个局部节点和 1 个中心节点,通过网络形成一个层次式的大数据挖掘构架.局部节点主要负责本节点的流式数据的收集和局部挖掘、中心节点则负责全局模式的挖掘.给定一个挖掘

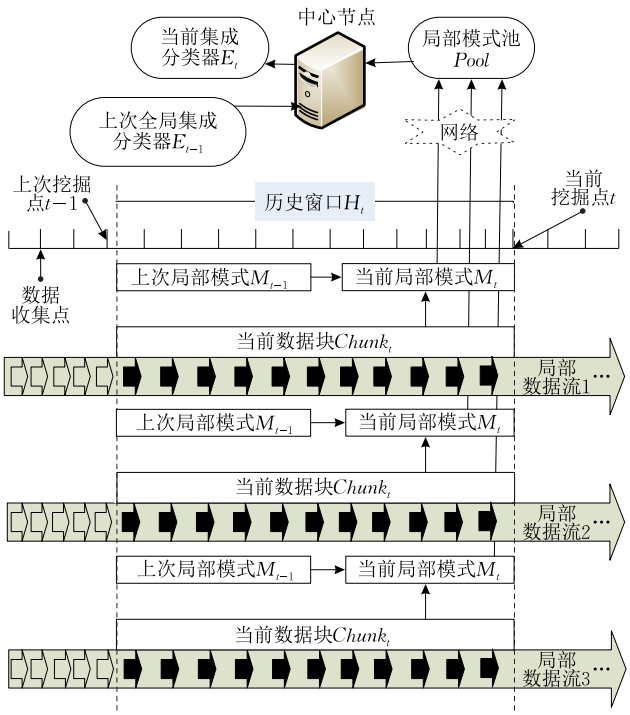


图 1 分布式、流式大数据的挖掘流程示意

时间点 t,分类挖掘的任务就是利用 t 时刻的时间窗口来将局部的微簇模式和全局的集成分类器更新到 t 时刻的(当前)状态.

如图 1 所示,一个大数据的分类挖掘有 3 个相对独立的阶段:

(1)局部挖掘,即在每个局部节点依据定义 1 的数据模型来收集当前数据块 $chunk_t$,然后利用 $chunk_t$ 对上一个挖掘点所维护的局部微簇模式进行增量式更新、形成新的(当前)微簇模式.

(2)模式传输,即当一个局部节点的微簇模式更新完成后就通过网络把它传送到中心节点.

(3)全局挖掘,即当所有局部节点的当前微簇模式都被成功地送到中心节点后,中心节点就进行全局集成分类器的学习,将全局模式更新到当前状态.

基于上面的定义和分析,定义 4 给出了对应的大数据的分类挖掘模型.

定义 4(大数据分类模型). 具有分布式和流动性为主要技术特征的大数据的分类模型定义为 $M=\langle T,D,O,P\rangle$,其中:$T=\langle t_1,t_2,\cdots\rangle$ 是收集数据的时间点序列;$D=\{S_1,S_2,\cdots,S_n\}$ 是依据 T 在局部节点上收集的 n 条局部数据流组成的分布式数据流,是数据挖掘的数据源;O 是对 D 的操作算子集,需要相应的算法来实现;P 是全局分类器,是学习的最终结果. 给定挖掘点 $t\in T$,具有分布式和流动性为主要技术特征的大数据的分类模型通过如下的操作来层次式构造:

(1)局部挖掘器,包含如下操作:

① $chunk\text{-}maker: D \rightarrow \{chunk_1, chunk_2, \cdots, chunk_t, \cdots\}$. 按照预先设定的挖掘点来收集窗口数据,$chunk_t$ 被称为当前数据块.

② $micro\text{-}cluster\text{-}abstractor: chunk_t \rightarrow \{m_1, m_2, \cdots, m_k\}$. 负责从当前数据块中挖掘出微簇集 $\{m_1, m_2, \cdots, m_k\}$.

③ $micro\text{-}cluster\text{-}maintainer: \{M_{t-1}; m_1, m_2, \cdots, m_k\} \rightarrow M_t$. 利用当前的微簇集来增量式更新上个挖掘点存储的微簇集 M_{t-1},得到当前挖掘点的微簇集 M_t.

(2)全局挖掘器,包含如下操作:

① $micro\text{-}cluster\text{-}pool: \{M_t\} \rightarrow pool$. 在中心节点,设置内存缓冲池 $pool$ 以收集所有局部节点发送来的局部微簇集.

② $sample\text{-}remaker: \{M_t\} \rightarrow S_t$. 将当前微簇集转换成全局学习样本集 S_t.

③ $ensemble\text{-}updater: \{E_{t-1}; S_t\} \rightarrow E_t$. 利用样本集 S_t 来增量式更新上个挖掘点的全局集成分类器 E_{t-1},得到当前集成分类器 E_t.

定义 4 的模型系统化地界定了具有分布式和流动性为主要技术特征的大数据的分类挖掘系统的主要功能需求及需要探索的主要挖掘操作算子.

(1)在局部挖掘器中,基本的操作算子包括当前数据块的获取、局部微簇模式挖掘以及局部微簇模式的更新维护等操作. 即:按照块到块的挖掘技术,通过调用局部挖掘器,在一个新数据块到达后,可以实现局部微簇模式的增量式更新.

(2)在全局挖掘器中,基本的操作算子涉及到局部微簇模式的(缓冲)存储、全局训练样本数据的生成(恢复)以及全局集成分类器的更新维护等操作. 即通过集成学习技术,当一个历史窗口的所有节点的局部模式到达中心节点后,通过触发全局挖掘器可以实现全局模式的增量式更新.

当然,在定义 4 模型中的挖掘操作算子需要通过设计对应的算法来实现,并且它们构成了分布式、流式大数据分类挖掘目标对应的核心算法,因此本文接下来的主要工作就是设计和分析这些核心算法.

4 算法设计和分析

本节将对定义 4 分类挖掘模型中的关键操作算

子进行算法设计. 主要的算法包括：在局部节点中的微簇抽取和增量式微簇维护算法；在中心节点中的学习样本数据重构和集成分类算法.

4.1　局部节点的微簇抽取算法

当一个局部节点的当前数据块被收集完成后，接下来的工作就是对其进行微簇挖掘. 首先，要对当前数据块进行类簇划分. 考虑到大数据的分类挖掘特点，可能存在类标识不全的情况，所以使用经典的无监督学习算法 k-means 进行聚类. 然后，对聚类得到的每个簇进行微簇抽取. 这个工作主要是依据定义 3 描述的微簇结构对每个类簇的数据进行对应的统计值抽取. 算法 1 给出了在局部节点中抽取微簇模式的过程描述.

算法 1.　*micro-cluster-abstractor*.

输入：当前挖掘时间点 t；在 t 时刻获得的数据块 D；数据维度 d；微簇个数 k

输出：在 t 时刻的微簇集合 M

1. divide D into k clusters C by Algorithm k-means;
2. FOR each $p\in C$
3. $p.n\leftarrow|p|$;
4. FOR $i=1$ to d
5. $p.c^i\leftarrow\left(\sum_{q\in p}q^i\right)\Big/n$;
6. $p.s^i\leftarrow\sqrt{\sum_{q\in p}q^i\times q^i}$;
7. $p.d^i\leftarrow\left(\sum_{q\in p}(q^i-p.c^i)^2\right)\Big/n$;
8. ENDFOR
9. $p.c\leftarrow(p.c^1,p.c^2,\cdots,p.c^d)$;
10. $p.s\leftarrow(p.s^1,p.s^2,\cdots,p.s^d)$;
11. $p.d\leftarrow(p.d^1,p.d^2,\cdots,p.d^d)$;
12. flag $p.f$ with the most label in instances of p;
13. integrate $p.n,p.c,p.s,p.d$ and $p.f$ into micro-cluster m;
14. $M\leftarrow M\cup\{m\}$;
15. ENDFOR
16. return M.

算法 1 中的步骤 1 是使用 k-means 进行聚类，其时间复杂度为 $o(l\times m)$，其中 l 和 m 是寻找最优划分簇的迭代次数和被挖掘的数据点大小[8]. 步骤 2～15 是实现微簇模式的构建，它的时间复杂度是 $O(k\times n)$，其中 k 和 n 分别是 k-means 挖掘出的类簇数目和每个簇中数据点的平均大小. 很显然，上面的 $m=k\times n$，所以算法 1 的时间复杂度由 k-means 算法的时间复杂度决定.

此外，算法 1 的内存空间占用情况是：除了 k-means 执行所必须的内存空间外，算法 1 的主要额外空间是为 k 个簇对应的微簇统计信息提供内存的数据结构. 按照定义 3，每个微簇结构只需要 5 个数值级别的存储空间. 因此，相对于传统的聚类算法 k-means，算法 1 只有很少的额外时间花费和空间消耗.

4.2　局部节点的增量式微簇维护算法

随着挖掘时间点的变化，一个局部节点维护的微簇集合需要及时更新来适应数据的变化. 依据图 1 所示的增量式方法，局部节点的微簇维护意味着：利用当前数据块获得的微簇集合 M^*，对上次挖掘点维护的微簇集合 M 进行增量式更新.

值得注意的是，作为局部节点的局部模式，一个节点上维护的微簇模式中的微簇数目必须适当加以控制，不能随着时间的增长无限制地膨胀. 这可以通过设置一个阈值参数 L 来加以控制. 当增量式更新导致微簇数目超过 L 时，就需要进行微簇合并. 因此，两个微簇的合并操作应该作为一个基本操作来加以研究，其中一个重要的问题就是如何在一个微簇集合中寻找两个最佳的微簇进行合并.

考虑分布式大数据的特点，我们采用“方差和最小”作为寻找最佳合并簇的标准. 即：给定一个微簇集合 M，被选择的用于合并的两个微簇 m_1 和 m_2 需要满足

$$\min\left\{\sum_{i=1}^{d}unite(m_1,m_2).d^i\,|\,m_1,m_2\in M\right\}\tag{4}$$

其中：$*.d^i$ 是一个微簇的方差的第 i 个维度值；$unite(m_1,m_2)$ 代表 m_1 和 m_2 合并后的微簇.

由于微簇并不是直接保存数据点，所以合并后的微簇模式不能通过定义 3 直接获得. 唯一的方法就是从两个待合并的微簇推导出合并后的微簇的统计值. 定义 5 给出了对应的计算方法.

定义 5（微簇的合并操作）.　给定两个 d 维的微簇 m_1 和 m_2，假如它们有共同的类标识，那么它们可以通过一个被称为合并的运算 $unite(m_1,m_2)$ 合并成一个新的微簇，记为 $m_3=unite(m_1,m_2)$，则 m_3 的统计值计算如下：

$$m_3.n\leftarrow m_1.n+m_2.n\tag{5}$$

$$m_3.c^i\leftarrow\frac{m_1.n\times m_1.c^i+m_2.n\times m_2.c^i}{m_3.n}\tag{6}$$

$$m_3.s^i\leftarrow\sqrt{(m_1.s^i\times m_1.s^i+m_2.s^i\times m_2.s^i)}\tag{7}$$

$$m_3.d^i\leftarrow\frac{m_1.s^i\times m_1.s^i+m_2.s^i\times m_2.s^i}{m_3.n}-2\times m_3.c^i\times\frac{m_1.c^i\times m_1.n+m_2.c^i\times m_2.n}{m_3.n}+m_3.c^i\times m_3.c^i\tag{8}$$

$$m_3.f \leftarrow m_1.f \tag{9}$$

定理 1. 设 $m_3=unite(m_1,m_2)$,则微簇 m_3 的统计值可以通过定义 5 的式(5)~(9)计算出,且计算结果正确反映合并前的数据分布.

证明. 不失一般性,假设 m_1 和 m_2 是一维的,且对应的原始数据集分别为 $c_1=\{x_1,x_2,\cdots,x_p\}$ 和 $c_2=\{y_1,y_2,\cdots,y_q\}$(因为是一维就不再标识维度).

设 $m_3=unite(m_1,m_2)$,则 m_3 对应的簇的原始数据集为 $c_1\cup c_2=\{x_1,x_2,\cdots,x_p,y_1,y_2,\cdots,y_q\}$.

根据定义 3 中微簇的点数计算,有

$$m_3.n=p+q=m_1.n+m_2.n \tag{10}$$

所以定义 5 的(5)成立.

根据定义 3 中微簇中心点的计算方法,有

$$\begin{aligned} m_3.c &=(x_1+\cdots+x_p+y_1+\cdots+y_q)/(p+q)\\ &=(x_1+\cdots+x_p+y_1+\cdots+y_q)/m_3.n \end{aligned} \tag{11}$$

$$\begin{aligned} m_1.c &=(x_1+x_2+\cdots+x_p)/m_1.n,\\ m_2.c &=(y_1+y_2+\cdots+y_q)/m_2.n \end{aligned} \tag{12}$$

应用式(12)到式(11),得到

$$m_3.c=(m_1.c\times m_1.n+m_2.c\times m_2.n)/m_3.n \tag{13}$$

所以定义 5 的式(6)成立.

根据定义 3 中微簇的平方和统计值计算,有

$$m_3.s=\sqrt{x_1^2+\cdots+x_p^2+y_1^2+\cdots+y_q^2} \tag{14}$$

$$\begin{aligned} m_1.s &=\sqrt{x_1^2+x_2^2+\cdots+x_p^2},\\ m_2.s &=\sqrt{y_1^2+y_2^2+\cdots+y_q^2} \end{aligned} \tag{15}$$

应用式(15)到式(14),得到

$$m_3.s=\sqrt{m_1.s\times m_1.s+m_2.s\times m_2.s} \tag{16}$$

所以定义 5 的式(7)成立.

根据定义 3 中微簇的方差统计值计算,有

$$\begin{aligned} m_3.d &=\frac{\sum_{i=1}^{p}(x_i-m_3.c)^2+\sum_{i=1}^{q}(y_i-m_3.c)^2}{p+q}\\ &=\frac{\sum_{i=1}^{p}x_i^2+\sum_{i=1}^{q}y_i^2-2\times m_3.c\times\left(\sum_{i=1}^{p}x_i+\sum_{i=1}^{q}y_i\right)}{p+q}+\\ &\quad m_3.c^2 \end{aligned} \tag{17}$$

应用式(15)、(12)和式(10)到式(17),得到

$$\begin{aligned} m_3.d=&\frac{m_1.s\times m_1.s+m_2.s\times m_2.s}{m_3.n}-\\ &2\times m_3.c\times\frac{m_1.c\times m_1.n+m_2.c\times m_2.n}{m_3.n}+\\ &m_3.c\times m_3.c \end{aligned} \tag{18}$$

所以定义 5 的式(8)成立.

最后,显然有

$$m_3.f \leftarrow m_1.f \tag{19}$$

由上面的式(10)、(13)、(16)、(18)和式(19)知道,在一维情况下定理 1 是正确的. 多维情况只需要按每个维度计算即可. 证毕.

定理 1 确保了定义 5 的微簇合并方法是正确的.这样,当维护的微簇模式超过限定的数目时,就可以通过重复执行两个微簇的合并操作来减少微簇的数目.算法 2 给出了在一个局部节点上进行微簇的增量式维护的基本过程.

算法 2. *micro-cluster-maintainer*.

输入:当前挖掘时间点 t;从当前块抽取的微簇集合 M^*;数据维度 d;在上次挖掘点维护的微簇集合 M;局部节点最大可被维护的微簇数目 L

输出:t 时刻更新的微簇集 M

1. $M\leftarrow M^*\cup M$;
2. $L_M\leftarrow|M|$;
3. WHILE $L_M>L$ DO
4. $b\leftarrow$ the largest number of machine;
5. FOR each $m_1\in M$
6. FOR each $m_2\in M$
7. IF $\sum_{i=1}^{d}unite(m_1,m_2).d^i<b$ THEN
8. $s_1\leftarrow m_1$; $s_2\leftarrow m_2$;
9. $b\leftarrow\sum_{i=1}^{d}unite(m_1,m_2).d^i$;
10. ENDIF
11. ENDFOR
12. ENDFOR
13. $p\leftarrow unite(s_1,s_2)$;
14. $M\leftarrow M\cup\{p\}$; $M\leftarrow M-\{s_1\}-\{s_2\}$;
15. $L_M\leftarrow L_M-1$;
16. ENDDO
17. return M.

算法 2 的时间花费取决于微簇的合并次数.很显然,由于基于 k-means 的微簇抽取算法将从当前数据块中产生 k 个微簇,所以算法 2 最多执行 k 次微簇的 *unite* 操作.但是,为了寻找这 k 次合并的微簇,需要按照式(4)进行测试.每次测试的微簇集合的容量和 L 相当,且在一个 L 大小的集合中进行两两测试的时间复杂度是 $O(L^2)$.因此,算法 2 的总的时间复杂度是 $O(k\times L^2)$.考虑到 k 和 L 都可以控制在合理范围内,因此算法 2 的时间效率可以得到保证.

此外,算法 2 的主要内存占用是 $k+L$ 个微簇对应的数据结构,所以算法 2 也不会产生过大的内存消耗.

4.3 中心节点的样本重构算法

依据图 1 的挖掘流程,在一个挖掘点上,当一个

局部节点的微簇模式被更新完成后，就会把它通过网络传输到中心节点. 当所有的局部节点的当前微簇模式都被传送到中心节点的缓冲池后，中心节点就会启动全局模式挖掘工作. 按照定义 4 给出的模型，为了提高全局模式的预测能力和抗干扰性，我们使用集成分类器作为全局模式. 这样，一个具有挑战性的问题就被提出：微簇模式不可能直接作为学习样本被使用，那么如何在中心节点获得集成学习所需的训练样本集就成为一个关键问题. 一个可行的方法就是利用局部节点传送过来的微簇模式来重新构造全局学习样本.

算法 3 给出了生成全局训练样本数据集的对应算法的伪代码.

算法 3. *Sample-remaker*.

输入：当前挖掘时间点 t；t 时刻从所有局部节点传送来的微簇集合 M；数据维度 d

输出：t 时刻重构的样本数据集 S

```
1.  FOR each m∈M
2.    n←m.n;
3.    FOR i=1 to n
4.      FOR j=1 to d
5.        r←rand(-1.1); 生成(-1,1)中的随机数;
6.        l← √(3×n×m.d^j/2);
7.        x^j←m.c^j+l×r;
8.      ENDFOR
9.      x←(x^1,x^2,…,x^d); 合成多维数据点 x;
10.     flag x with m.f;
11.     insert x into S;
12.   EDNFOR
13. ENDFOR
14. return S.
```

很显然，算法 3 的时间复杂度（不考虑数据维度）是 $O(n)$，其中 n 是恢复的样本数目. 内存消耗也主要是 n 个样本数据所需的空间.

理论上说，重构的训练样本集和原始数据集必须是等价的，至少应该保持重要的统计参数值. 定理 2 从理论上保证了这点，因此算法 3 采用的数据恢复方法是合理的.

定理 2. 假如分布式数据流中的数据满足正态分布，则对于每个微簇来说，算法 3 重新构造的数据集与原始的微簇的均值和方差统计值是等价的.

证明. 不失一般性，假设数据是一维的，处理的微簇集合 M 只有一个微簇 m，m 对应的均值和方差分别是 μ 和 σ；对 m 实施算法 3 后得到了样本集 $X=\{x_1,x_2,\cdots,x_n\}$.

根据微簇的均值定义，计算 X 的中心点 X_c：

$$X_c=1/n\times\sum_{i=1}^{n}x_i \tag{20}$$

根据算法 3，X 的点产生如下：

$$\forall x_i\in X, x_i\leftarrow\mu+\sqrt{3n\sigma/2}\times rand(-1,1) \tag{21}$$

代入式(21)到式(20)，有

$$X_c=1/n\times\sum_{i=1}^{n}(\mu+\sqrt{3n\sigma/2}\times rand(-1,1))$$

$$=\mu+1/n\times\sqrt{3n\sigma/2}\times\sum_{i=1}^{n}rand(-1,1) \tag{22}$$

因为正态分布下，$\sum rand(-1,1)\sim\int_{-1}^{1}x\,\mathrm{d}x=0$，所以根据式(22)，有

$$X_c\sim\mu \tag{23}$$

另外，根据定义 3 的方差定义，X 的方差 X_d 为

$$X_d=1/n\times\sum_{i=1}^{n}(x_i-X_c)^2 \tag{24}$$

代入式(21)到(24)，得到

$$X_d=1/n\times\sum_{i=1}^{n}(\mu+\sqrt{3n\sigma/2}\times rand(-1,1)-X_c)^2$$

$$=1/n\times\sum_{i=1}^{n}((\mu-X_c)+\sqrt{3n\sigma/2}\times rand(-1,1))^2 \tag{25}$$

根据式(23)的 $X_c\sim\mu$，式(25)和下面式子等价：

$$X_d\sim\frac{\sum_{i=1}^{n}(\sqrt{3n\sigma/2}\times rand(-1,1))^2}{n}$$

$$=\frac{3n\times\sigma\times\sum_{i=1}^{n}rand(-1,1)^2}{2n}$$

$$=3\sigma/2\times\sum_{i=1}^{n}rand(-1,1)^2 \tag{26}$$

因为 $\sum rand(-1,1)^2\sim\int_{-1}^{1}x^2\,\mathrm{d}x=2/3$，所以，有

$$X_d\sim\sigma \tag{27}$$

上面式(23)和式(27)说明在 1 个微簇和 1 维数据空间的情况下，定理 2 成立. 当多个微簇或者多维数据时，只需要对每个微簇或者每个维度使用上面方法进行推理. 证毕.

4.4 中心节点的集成分类器更新算法

集成分类器的构造首先需要选择一个基础（弱）分类器. 本文选用 C4.5，它是经典的高效数据分类算法[21]，特别是由于它良好的剪枝和优化机制能很好地适应大数据的处理.

此外，集成分类器的更新策略也是一个重要的问题. 目前流行的集成分类器大多都是基于 Boosting

和 Bagging 模型的[21]. 基于 Boosting 的模型需要不断学习与更新样本数据和弱分类器的权重,对于大数据来说不仅计算太复杂、而且对于像 C4.5 这样的决策树模型来说也缺乏足够的稳定性[20]. 基于 Bagging 的模型随机选取每个弱分类器的训练样本,对弱分类模型要求不高,但它的收敛速度要比基于 Boosting 模型差. 考虑分布式、流式大数据的特点,特别是对全局模式的高归纳性及抗干扰性的要求,我们将借鉴 Boosting 和 Bagging 技术,力求集成两者的技术优势来设计本文的全局集成分类器.

本文集成学习策略可以简单地归纳为学习样本的淘汰策略. 主要思想是:给定一个集成分类器 $E=\{e\}$,当一个学习样本 s 在一个弱分类器中被正确预测时,就及时将 s 淘汰掉、不再用它作为其他的弱分类器的训练样本. 这样做的目的是为了尽可能地保证弱分类器之间的差异以提高对样本的覆盖度,同时也提高了集成分类器的学习效率. 算法 4 给出了中心节点学习集成分类器的过程描述.

算法 4. *ensemble-updater*.

输入:当前挖掘时间点 t;训练样本集 S;在上个挖掘点维护的集成分类器 E;集成分类器中允许的最多弱分类器个数 Q

输出:t 时刻更新的集成分类器 E

```
1.  p←|S|/Q; 设定弱分类器的样本数
2.  FOR each e∈E
3.    e.error←0;
4.    FOR s∈S
5.      f←e(s); 使用弱分类器 e 进行预测
6.      IF f=s.f THEN delete s from S
7.      ELSE e.error←e.error+1;
8.    ENDFOR
9.    IF S=∅ THEN break
10.   ELSE
11.     K←randomly select p samples from S;
12.     e*←C4.5(K) ;学习一个新分类器
13.     E←E∪{e*};
14.   ENDIF
15.   IF |E|>Q THEN
16.     c←e that satisfies max{e.error|e∈E}
17.     E←E−{c};超过上界时删除最差的弱分类器
18.   ENDIF
19. ENDFOR
20. return E.
```

算法 4 在每个挖掘时间点更新全局集成分类器. 依据增量式更新的策略,针对每个弱分类器,做了如下 3 个工作:

(1)步骤 3～8 对所有的训练样本针对一个弱分类器进行测试,对于测试正确的样本进行淘汰(即不用它们产生新的弱分类器). 假设整个训练样本集有 n 个数据点,则其时间花费和 n 成正比.

(2)步骤 9～14 利用未被淘汰的样本数据(随机选取)进行新的 C4.5 分类器学习. 假设 C4.5 的平均学习时间是 c 的话,那么步骤 9～14 的时间花费和 c 相当.

(3)步骤 15～18 完成一个弱分类器的裁剪(在弱分类器数目超过设定的阈值时),这需要在被观察的集成分类器中对每个弱分类器的错误情况进行比较,需要测试的次数和 Q 相当,其中 Q 是集成分类器中弱分类器数目的上限值.

根据算法 4,上面的 3 个工作需要对每个弱分类器进行,而弱分类器的数目由 Q 值决定,所以整个算法 4 的时间复杂度是 $\mathrm{Max}\{O(Q\times n),O(Q\times c),O(Q^2)\}$. 一般地说,$Q$ 是远小于 n 和 c,因此算法 4 的执行时间主要取决于训练数据的规模和 C4.5 算法的训练学习时间.

算法 4 的内存消耗主要是集成分类器中所有 C4.5 对应的数据结构,与集成分类器的弱分类器数目有关.

5 实验与分析

为了评估本文模型和算法的有效性,我们使用公共数据集 KDD(CUP)99① 来构建对应的大数据挖掘的训练和测试数据. KDD99 的原始数据来自于 MIT 林肯实验室收集的美国空军模拟网的流量监控数据,后来被哥伦比亚大学等整理成规范的公共数据集. KDD99 数据集是网络连接记录的时间序列数据,所以被认为是研究流式的网络流量数据分析及入侵检测模式评价的标尺数据集(benchmark). 同时,由于它的训练数据部分已经被很好地标注,而且网络流量及入侵检测分析的主要手段是分类挖掘,因此也被用来评估数据流中的分类算法. KDD99 的整个训练数据有 5 000 000 多个网络连接记录,有 41 个学习的条件属性,包含正常(normal)或攻击(attack)两大类,后者又被分级成 4 个二级类别. 考虑到本文主要是进行分类挖掘(而不是专门的入侵检测)实验,因此本文的实验不再对 4 个二级类别做

① KDD Cup 1999 Data (KDD Cup 99). http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

进一步划分和区分.

为了模拟分布式、流式大数据，我们编制了一个数据流生成的软件工具 stream-producer. 它通过对 KDD99 数据集的 I/O 操作来模拟流动数据的在线到达情况. stream-producer 主要通过数据收集时间间隔和数据流速两个参数来控制数据流. 本文实验的设置为：数据收集时间间隔是 1/2000 s；流速范围 1000～2000 记录/s. 设置不同的数据流速是为了更好地模拟数据的随机流动情况，所以实验是针对不同的时间段内收集的数据容量可能不同的实际情况设计的.

本文实验基于 3 个局部节点和 1 个中心节点的分布式数据流应用环境，使用 4 台 Intel 酷睿 i7、内存 2 GB 的计算机构成对应的硬件单元. 利用 Hadoop 的 HDFS 分布式文件系统，将 KDD99 数据集分布存储在 3 个局部节点中，中心节点作为 Master 节点，负责对应的文件目录信息的维护. 在每个局部节点上，stream-producer 工具被部署，来模拟数据流的产生过程以形成窗口数据.

算法 1～4 主要采用 Hadoop 的 MapReduce 编程方法来实现. 例如，就局部微簇的抽取(算法 1)而言，*Map*()函数的键值对被设计成〈数据元组，簇号〉，完成数据块的 *k*-means 聚类映射操作，而对应的一次 *Reduce*()函数的调用则完成一个数据簇(相同簇号)的统计计算，即微簇模式抽取. 按照定义 4 模型和图 1 的流程，局部挖掘器的核心处理是基于算法 1 和 2 的，全局挖掘器的核心处理是基于算法 3 和 4 的. 因此，在每个局部节点，需要以算法 1 和 2 为核心形成完整的局部挖掘器，在中心节点以算法 3 和 4 为骨干形成完整的全局挖掘器.

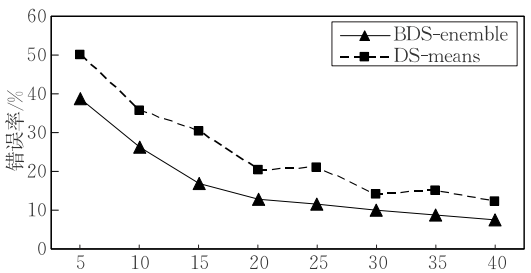
实验中通过增加“本文算法的连接程序模块”，实现了完善的局部挖掘器和全局挖掘器功能. 例如，就局部挖掘器而言，通过算法 1 对应的 Reduce 函数的调用可以得到所有数据簇对应的微簇集合，所以在微簇合并操作(算法 2 的核心操作)的实现中，我们设计的 *Map*()键值对是〈(簇号 1，簇号 2)，方差和〉，调用它来完成一个微簇集的方差和映射，其中输入的簇号 1 和 2 都来自于算法 1 中 Reduce 的输出结果. 对应的微簇合并中的 *Reduce*()的功能则是寻找最小方差和、并将最小方差和的两个微簇合并起来以实现对应的微簇合并功能. 类似的，全局挖掘器也通过这样的编程方式加以实现. 就局部挖掘器和全局挖掘器的连接问题而言，本文实验是通过中心节点的存储缓冲值的变化来触发的，即当 3 个局部节点的局部微簇模式全部到达中心节点时就触发全局分类器开始工作.

正如定义 4 模型所刻画的那样，以分布式和流动性为主要技术特征的大数据的分类挖掘是在多个节点、由多个步骤来协作完成的，以全局分类器为最终的挖掘成果，因此全局分类器的精度决定一个方案的整体挖掘精度. 但是，一个分布式系统的挖掘时间和内存消耗总是被分散在多个节点上，因此依据定义 4 模型，局部挖掘器和全局挖掘器都有对应的时空效率评估问题. 当然，全局的性能评估是最重要的，所以以下的实验主要是针对全局分类器的性能进行评价.

本文方法简称为 BDS-ensemble，使用的主要控制参数有 *H-size*、*unlab*%和 *E-no*，它们分别代表时间窗口大小、未标签的样本比例和集成分类器的弱分类器数目.

对比算法是 DS-means[8]，其中 DS-means 的类簇数目是根据数据容量动态产生的(按照原始论文方法). 在本文实验中，DS-means 产生的类簇数目在 30～100 的范围内. 之所以选择 DS-means 作为对比算法，主要是考虑它和本文方法都是基于局部节点和中心节点构成的层次式挖掘构架的，因此具有可比性.

实验 1(不同历史窗口下的精度测试). 当固定其他控制参数时，按照定义 4 模型及相关的算法 1～4，在 1000 s 内的训练数据流中，使用本文方法 BDS-ensemble 和对比算法 DS-means 生成对应的全局分类器. 然后，利用 KDD99 中的测试数据集来进行精度测试. 图 2 给出了随着历史窗口长度增加时 BDS-ensemble 和 DS-means 的错误率对应变化.



(固定*unlab*%=25%、*E-no*=20，观察不同*H-size*下错误率变化)

图 2　时间窗口长度增加时错误率的变化

图 2 表明，BDS-ensemble 的错误率明显小于 DS-means(平均优于 10%左右). 因为历史窗口变长意味着收集的数据块变大，所以在 1000 s 处理的窗口数目变少，因而挖掘精度应该被期望变高. 图 2 说明，当历史窗口增大时 BDS-ensemble 的精度在

逐步提升,而 DS-means 则表现出波动.当然,设置窗口的目的就是为了利用有限的计算机资源(内存、CPU 等)来解决潜在无限的大数据问题,所以也不可能通过无限度地增加窗口长度来换取分类精度.

另外从图 2 中也可以看出,相比 DS-means,BDS-ensemble 的稳定性要好.就图 2 而言:当窗口长度达到 30s 时,BDS-ensemble 的错误率已经下降到 10%左右,当再加大窗口长度时,错误率的下降幅度明显收窄.这种稳定性使得代价与精度的优化的分布式流式解决方案成为可能.

实验 2(不同的集成分类器设置下的分类精度测试). 固定 *H-size* = 30 和 *unlab*% = 25%,在 1000s 内,每增加 100s 执行 1 次 BDS-ensemble 和 DS-means 来生成全局分类器,然后进行错误率测试.本实验的目的是测试集成分类器中的弱分类器数目对挖掘精度的影响,图 3 给出了不同的弱分类器数目下的 BDS-ensemble 的错误率随着训练时间增加时的变化情况(对比算法是 DS-means).此外,为了更清楚地看到两种方法的差别,图 4 把 *E-no* = 20 时的 BDS-ensemble 和 DS-means 的错误率用折线图形式展示出来.

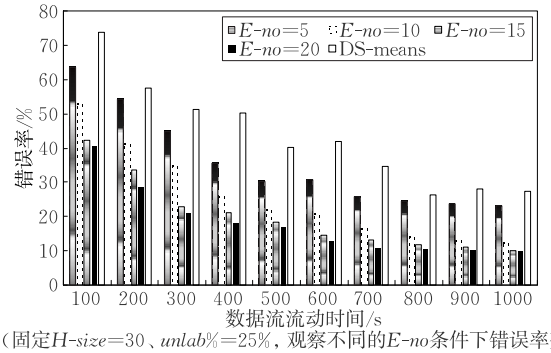
从图 3 中可以看出:当 *E-no* = 5 时,BDS-ensemble 的分类精度并不是很好;在 *E-no* = 10 时已经有很大改善;当 *E-no* = 15 和 *E-no* = 20 时,BDS-ensemble 的分类精度随着训练时间的增加在稳步提升,当训练了 1000s 后,其分类精度已经达到 90%左右.相比而言,DS-means 则不稳定,而且分类精度明显要比本文方法要差.

从图 4 中可以清楚看到:当集成分类器维持一定大小的弱分类器数目时(如实验中的 *E-no* = 20),BDS-ensemble 会随着训练时间的增加分类精度逐步提升,而且逐步稳定在一定的精度范围内.相比较而言,DS-means 则存在一定的波动.

实验 3(执行时间和内存空间测试). 固定 *unlab*% = 25% 和 *E-no* = 20,通过设置不同的历史窗口长度,跟踪 BDS-ensemble 和 DS-means 执行时在中心节点的时间花费和空间消耗.因为 BDS-ensemble 和 DS-means 都需要依次经过局部节点和中心节点挖掘来进行,在相应节点上都有对应的 CPU 时间占用和内存空间消费,因此本实验只跟踪了中心节点的时间和空间消耗.

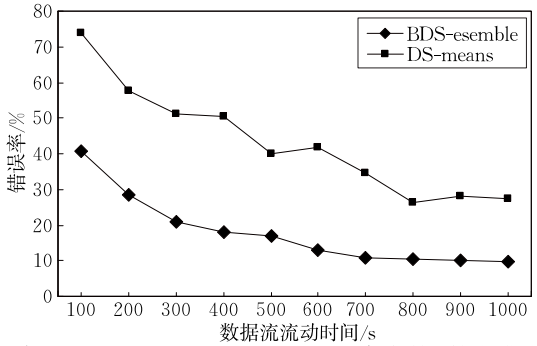
就中心节点而言,它们的时间和空间消耗主要花费在全局模式的更新上.图 5 给出了在中心节点中 BDS-ensemble 和 DS-means 学习全局分类器的时间花费比较,而图 6 则给出了 BDS-ensemble 和 DS-means 学习全局分类器的内存消耗情况.

从图 5 中可以看出,BDS-ensemble 和 DS-means 随着历史窗口的时间增长,一次增量式更新的时间



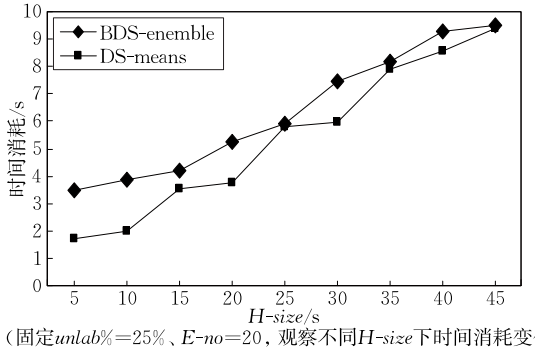
(固定*H-size*=30、*unlab*%=25%,观察不同的*E-no*条件下错误率变化)

图 3 流动数据在不同的集成分类设置下的错误率变化



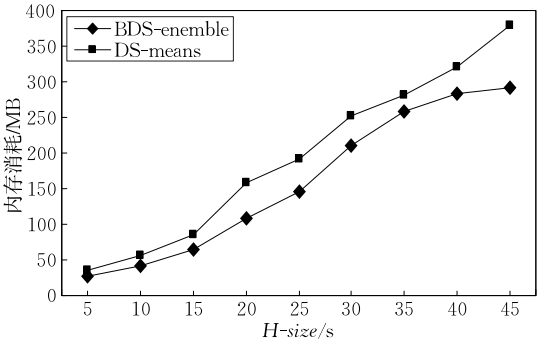
(固定*H-size*=30,*unlab*%=25%,*E-no*=20,观察随时间增长的错误率变化)

图 4 *E-no* = 20 时错误率变化的情况



(固定*unlab*%=25%、*E-no*=20,观察不同*H-size*下时间消耗变化)

图 5 时间窗口长度增加时执行时间的变化情况



(固定*unlab*%=25%、*E-no*=20,观察不同*H-size*下内存消耗变化)

图 6 时间窗口长度增加时内存空间的变化情况

花费都会增加.这是因为历史窗口增长使得 CPU 的窗口计算时间增加.同时,虽然 BDS-ensemble 比 DS-means 略高,但是两者的攀升幅度相当.这主要是因为 BDS-ensemble 使用的集成学习方法需要多次调用 C4.5 算法.但是,按照算法 4 的设计,每次调用 C4.5 算法时不是使用全部的训练样本、而是随机选取部分样本,所以 BDS-ensemble 的执行时间并没有比 DS-means 有特别明显地提升.

图 6 中说明,BDS-ensemble 和 DS-means 的内存消耗相差不大.同样是由于 BDS-ensemble 在调用 C4.5 算法时使用地是部分训练样本,所以内存消耗要稍微少于 DS-means.

综合实验 1、2 和 3,面对复杂的大数据应用,本文方法能够通过较小时间代价的增加获得较大挖掘精度的提升,进而在精度与代价的平衡上获得了一个较优化的解决方法.

6　优势和局限性讨论

本文针对大数据 4V 属性难以进行形式化研究的问题,分析了大数据隐藏的技术属性.从概念上讲,不同技术属性及其组合能演变出特定类型的大数据形态.聚焦在大数据的分布式和流动性的技术特征上,本文研究了这类大数据的知识挖掘问题,并且可以契合电子商务业务、股票交易等大数据的应用需求.

本文模型和方法的优势首先表现在合理的网络通讯代价上.大数据的分布式挖掘必须面对代价与精度的平衡问题,再高精度的方法如果存在无法承受的代价(包含网络传输量)都是不可行的.本文方法的优势也体现在方法和技术的集成上.大数据中分布式挖掘问题应该建立在多节点、多步骤协同工作基础上,而不同数据挖掘方法在不同挖掘步骤中的有效性和局限性是客观存在的,因此针对不同步骤或者阶段来寻找和实施对应的挖掘技术是本文解决问题的基本理念之一.本文在局部节点采用微簇挖掘技术、在全局节点利用集成学习方法、针对大数据特点来研究和使用块到块的增量式挖掘手段等,都体现了这种理念的运用.对应的实验也验证了这种集成化的解决方案的有效性和科学性.

当然,本文模型和方法也存在应用的局限性.主要表现在两个方面:(1)本文探讨的只是大数据中的一类数据形态,并不能完全适应所有的大数据应用.例如:对于像大型电子商务网站的交易大数据而言,本文模型和方法可以提供一种可行的解决方案;但是对于像门户网站这样具有数据类型多样化的大数据而言,基于本文定义 1 的数据形态就无法形成规格化的数据描述,因而本文的模型和方法就不能完全解决这类问题;(2)本文探讨的全局样本恢复算法是以数据的正态分布为前提的,对于数据分布极其不规范的情况,算法的精度可能会下降.

7　总　结

大数据概念在强大的应用需求下被提出,而且随着云计算等软硬件基础设施的发展,使得大数据的分析成为可能.然而,大数据的研究和应用仍然处于起步阶段,有许多挑战性的问题需要逐步解决.特别是,在理论和方法上的需求越来越迫切.本文从大数据的分布式和流动性这些技术特征出发,在挖掘模型及关键算法等方面进行了系统化地研究和设计.

本文首先从大数据的应用需求入手,分析了具有分布式和流动性技术特征的大数据的应用范围和潜在的应用价值.然后,借助于分布式数据流概念刻画了这类大数据的抽象数据模型,并在此基础上设计了一个大数据的分类模型.最后,对于模型中关键操作对应的算法进行了设计.本文对应的研究,突出体现了利用技术的集成化来系统化地解决大数据挖掘问题的主要思想,在分布式的微簇挖掘、块到块的增量式学习以及基于淘汰策略的集成分类器学习等理论和方法上进行了创新性的工作.本文对应的大数据的分类挖掘构架及其系列算法,不仅可以全方位地改善以分布式和流动性为主要技术特征的大数据的分类挖掘效果,而且在分布式计算、内存占用及节点间的网络通讯代价的平衡问题上可以获得了一个优化的结果.

进一步的工作包括:针对非正态化的数据分布情况,研究基于密度估计等的样本重构方法;寻找非样本重构技术的全局模式挖掘方法;针对大数据的其他需求,研究分类以外的挖掘问题,如关联规则、概念归纳等;针对大数据的其他技术特征,如高维性、数据半结构化等,开展相应的理论、模型和算法研究.

参　考　文　献

[1] James M, Michael C, Brad B, et al. Big data: The next frontier for innovation, competition and productivity. McKinsey Global Institute, Silicon Valley, USA: Technical Report 0983179697, 2011

[2] Viktor Mayer-Schönberger. Big Data: A Revolution That Will Transform How We Live, Work and Think. Boston, USA: Houghton Mifflin Harcourt, 2013(in Chinese)
(盛杨燕，周涛译. 大数据时代. 杭州：浙江人民出版社，2013)

[3] Li Xue-Long, Gong Hai-Gang. Survey on big data system. Science China Information Sciences, 2015, 45(1): 1-44(in Chinese)
(李学龙，龚海刚. 大数据系统综述. 中国科学：信息科学，2015，45(1)：1-44)

[4] Guo Hua-Dong, Wang Li-Zhe, Chen Fang, et al. Scientific big data and digital earth. Chinese Science Bulletin, 2014, 59(12): 1047-1054(in Chinese)
(郭华东，王力哲，陈方等. 科学大数据与数字地球. 科学通报，2014，59(12)：1047-1054)

[5] Wang Yuan-Zhuo, Jin Xiao-Long, Cheng Xue-Qi. Network big data: Present and future. Chinese Journal of Computers, 2013, 36(6): 1125-1138(in Chinese)
(王元卓，靳小龙，程学旗. 网络大数据：现状与展望. 计算机学报，2013，36(6)：1125-1138)

[6] Sun Da-Wei, Zhang Guang-Yan, Zheng Wei-Min. Big data stream computing: Technologies and instances. Journal of Software, 2014, 25(4): 839-862(in Chinese)
(孙大为，张广艳，郑纬民. 大数据流式计算：关键技术及系统实例. 软件学报，2014，25(4)：839-862)

[7] Luo D, Ding C, Huang H. Parallelization with multiplicative algorithms for big data mining//Proceedings of the 12th IEEE International Conference on Data Mining (ICDM). Washington, USA, 2012: 489-498

[8] Guerrieri A, Montresor A. DS-means: Distributed data stream clustering//Kaklamanis C, Papatheodorou T, Spirakis P G eds. Euro-Par 2012 Parallel Processing. Lecture Notes in Computer Science 7484. Berlin, Germany: Springer, 2012: 260-271

[9] Cesario E, Mastroianni C, Talia D. A multi-domain architecture for mining frequent items and itemsets from distributed data streams. Journal of Grid Computing, 2014, 12(1): 153-168

[10] Wang E, Chen A. Mining frequent itemsets over distributed data streams by continuously maintaining a global synopsis. Data Mining and Knowledge Discovery, 2011, 23(2): 252-299

[11] Street W, Kim Y. A streaming ensemble algorithm (SEA) for large-scale classification//Proceedings of the ACM Conference on Knowledge Discovery and Data Mining(KDD01). San Francisco, USA, 2001: 377-382

[12] Phillip B, Srikanta T. Distributed streams algorithms for sliding windows. Theory of Computing Systems, 2004, 37(3): 457-478

[13] Yang Y, Mao G. A self-adaptive sliding window technique for mining data streams//Proceedings of the International Conference Information and Multimedia Technology. Hong Kong, China, 2010: 56-60

[14] Parthasarathy S, Ghoting A, Otey M. A survey of distributed mining of data streams//Aggarwal C C ed. Advances in Database Systems. New York, USA: Springer, 2007: 289-307

[15] Bhaduri K, Das K, Sivakumar K, et al. Algorithms for distributed data stream mining//Aggarwal C C ed. Advances in Database Systems. New York, USA: Springer, 2007: 309-331

[16] Anceaume E, Busnel Y. A distributed information divergence estimation over data streams. IEEE Transactions on Parallel and Distributed Systems, 2014, 25(2): 478-487

[17] Wu X, Zhu X Q, Wu G Q, Ding W. Data mining with big data. IEEE Transactions on Knowledge and Data Engineering, 2013, 26(1): 97-107

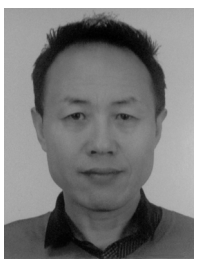
[18] Masud M, Gao J, Khan L, et al. A practical approach to classify evolving data streams: Training with limited amount of labeled data//Proceedings of the IEEE International Conference on Data Mining (ICDM). Pisa, Italy, 2008: 929-934

[19] Kranen P, Assent I, Baldauf C, Seidl T. The ClusTree: Indexing micro-clusters for anytime stream mining. Knowledge and Information Systems, 2011, 29(2): 249-272

[20] Prenger R, Lemmond T, Varshney K, et al. Class-specific error bounds for ensemble classifiers//Proceedings of the 16th ACM International Conference on Knowledge Discovery and Data Mining(KDD10). Washington, USA, 2010: 843-852

[21] Rokach L. Ensemble-based classifiers. Artificial Intelligence Review, 2010, 33(1): 1-39

[22] Li H, Huang H, Lee S. Fast and memory efficient mining of high-utility itemsets from data streams: With and without negative item profits. Knowledge and Information Systems, 2011, 28(3): 495-522



MAO Guo-Jun, born in 1966, Ph.D., professor. His research interests include data mining, big data and distribution computing.

HU Dian-Jun, born in 1989, M.S. candidate. His research interests include big data and data mining.

XIE Song-Yan, born in 1992, M.S. candidate. Her research interests include big data and data mining.

Background

Since the concept of big data was proposed, it has been concerned by public, academics and business people. However, it is a fact that more people are exploring application requires of the Big Data Time rather than studying technical methods of mining Big Data, and so such a current situation in Big data would not be propitious to its development over a long period of time.

Nowadays, with developments of Internet, the data related to the service of Internet are rapidly growing. For example, large electronic business websites like eBay can generate data of PB level per month, and such big data are often distributed in multiple computer nodes and streaming over time. Therefore, distributed streaming data are becoming an important character of big data on Internet. Similarly, such distributed streaming data can appear in computer network traffic, bank transactions and sensor network systems.

Aiming at mining the distributed streaming big data, many challenging theoretic and technical problems are coming. One of these challenges is how to construct an effective mining framework to make distributed computing possible when there are not too large network transferring costs to increase. Another key problem is how to efficiently get the global knowledge patterns from a distributed environment. Therefore, this paper discusses the basic technical requirements and effectively mining algorithms to distributed streaming big data.

The main research work includes several parts, which will be represented as below: (1) By analyzing the basic need to mining distributed streaming data, a classifying model is constructed to it; (2) Based on the mining model, a few of the key algorithms are designed, which are related to the key phases for mining a distributed streaming big data; (3) In theory, the rationalities of the key algorithms are partly proven, and a series of experiments are done which show that our solution and methods in this paper can get higher mining accuracies to the distributed streaming big data, while time and space consumptions to mine are kept in the appropriate ranges.

This paper is partly supported by the National Natural Science Foundation of China under Grant No. 62173293, which is planned to uncover some important mining principles and operating mechanics in distributed data streams. The authors have worked on data stream mining for many years, and recently paying a great effort on mining big data. The first author has published more than 100 papers in international or domestic conferences and journals.