DOI:10.13718/j.cnki.xsxb.2019.07.010

基于 Spark 框架的大数据 K-prototypes 聚类算法①

龚 静

铜仁学院 大数据学院,贵州 铜仁 554300

摘要:大数据具有数据量大及混合类型的属性,基于 MapReduce 的 K-prototypes 并行大规模混合数据方案的缺点是时间和内存的限制,导致这些方案不适合处理大数据.为了解决这个问题,该文提出一种新的基于 Spark 的 k-prototypes 聚类方法,该方法使用了重新聚集技术,利用 Spark 框架的内存操作来构建大规模混合数据分组.在模拟和实际数据集上的实验表明,该文方法可行,且提高了现有 K-prototypes 方法的效率.

关 键 词:大数据;混合数据;K-prototypes;Spark 框架

中图分类号:TP311 **文献标志码**:A **文章编号**:1000-5471(2019)07-0063-06

随着互联网技术的发展,各个领域海量数据已经成为常态,分析大数据的方法要求越来越高,因此使用机器学习技术探索大规模混合数据成为大数据分析中的一个重要挑战[1-2].鉴于大数据通常由混合类型的属性来描述,需要预处理步骤将数据转换为单一类型,因为大多数提出的聚类方法仅处理数值属性或分类属性.但是,转换策略通常耗时且会导致信息丢失,从而导致聚类结果不准确[3-4],传统聚类算法有 K-means 和 C-means 聚类算法等[5-6].

对于大数据处理方法,聚类方法已经得到研究[7],K-prototypes 聚类方法[8]中集成了 K-means 和 K-modes方法来聚类数值和分类数据.针对 K-prototypes 聚类方法的不足,改进 K-prototypes 聚类方法已经被提出用来处理混合类型的数据,如用于表示群集中分类属性原型的分布式质心概念的引入[9].虽然 K-prototypes方法是混合数据执行聚类最流行的方法之一,但该方法还是无法对大规模混合数据进行扩展.

为了处理大规模的数据,现有方法都是在并行框架上进行聚类,大多数方法都使用 MapReduce 框架,例如 MapReduce 框架实现的模糊 C-means 方法[10]、基于单通道和线性时间 MapReduce 的 k-means 方法[11]和基于密度聚类与 MapReduce 的并行化方法[12].虽然以上方法为用户提供了通过 MapReduce 框架对大规模数据的有效分析,但它们不能支持混合类型的数据,并且仅限于数字属性.因此,通过 MapReduce 框架下的 K-prototypes 方法能够并行化处理大规模混合数据[13],然而 MapReduce 框架与 K-prototypes 方法具有一个缺陷,在每次迭代过程中都会发生许多 I/O 磁盘操作,这会减慢运行时间.

为了解决这个问题,本文提出一种新的基于 Spark 框架的 K-prototypes 聚类方法,称为 S-KP.本文算法利用 Spark 提供的灵活性,通过减少内存操作来解决现有 MapReduce 方案的消耗时间.实验表明,所提出的方法是可扩展的,并且优于现有方法的效率.

1 大数据技术与 K-prototypes 聚类方法

1.1 大数据技术

MapReduce 是一个并行编程框架,旨在处理机群聚类中的大规模数据,其特点是对程序员高度透明,

① 收稿日期:2018-06-01
基金项目:贵州省教育厅普通高等学校创新人才团队建设项目(黔教合人才团队字[2015]67 号).
作者简介:龚 静(1974-),女,硕士,教授,主要从事计算机网络、数据挖掘及教育信息技术研究.

允许以简单和舒适的方式并行化算法，要并行化的算法只需要指定2个阶段即map和reduce. 每一个阶段中都有<*key*/*value*>作为输入和输出，map阶段并行采用每个<*key*/*value*>对并生成一组中间<*key'*/*value'*>对，然后该框架将与相同中间<*key*/*value*>对相关联的所有中间值与列表(称为shuffle阶段)进行分组，reduce阶段将此列表作为生成最终值的输入，MapReduce的输入和输出存储在一个关联的分布式文件系统中，该系统可以从使用聚类的任何一台机器访问. 根据可用的聚类体系结构，MapReduce框架的不同实现是可能的.

MapReduce可以在Apache Hadoop上实现，Apache Hadoop是用于商业硬件上大数据处理和存储最受欢迎的MapReduce实现，尽管Hadoop MapReduce非常流行，但在迭代算法中存在问题.

Apache Spark是一种新的大数据处理框架，旨在解决Hadoop的缺点. 该框架提出了一套超越标准MapReduce的内存转换，目的是在分布式环境中更快地处理数据，速度比Hadoop快100倍. Spark基于弹性分布式数据集(RDD)，这是一种用于以透明方式执行并行计算的特殊类型的数据结构. 这些结构持久存储、重用和缓存结果. 此外，还管理分区以优化数据放置并使用一系列广泛的转换操作数据，Spark框架的这些功能使其成为大数据处理的有用框架.

1.2 K-prototypes聚类方法

给定包含n个数据点的数据集$X=\{x_1, x_2, \cdots, x_n\}$，用数据属性$m_r$和分类属性$m_t$来描述. K-prototypes聚类的目的是通过最小化目标函数来找到k个聚类.

$$J=\sum_{i=1}^{n}\sum_{j=1}^{k}u_{ij}d(x_i, c_j) \tag{1}$$

其中，$u_{ij}\in\{0, 1\}$是分区矩阵U_{n*k}的元素，表示数据点i在簇j中的隶属度，$c_j\in C=\{c_1, \cdots, c_k\}$为聚类$j$的中心，$d(x_i, c_j)$是不相似度度量，定义为

$$d(x_i, c_j)=\sum_{r=1}^{m_r}\sqrt{(x_{ir}-c_{jr})^2}+\sum_{t=1}^{m_t}\delta(x_{it}, c_{jt}) \tag{2}$$

x_{ir}表示数字属性r的值，x_{it}表示数据点i的分类属性t的值，c_{jr}表示数值属性r和簇j的平均值，计算得

$$c_{jr}=\frac{\sum_{i=1}^{|c_j|}x_{ir}}{|c_j|} \tag{3}$$

其中，$|c_j|$表示分配给簇j的数据点数量，c_{jt}表示分类属性t和簇j的最常见值，计算方式为

$$c_{jt}=a_t^h \tag{4}$$

其中，$f(a_t^h)\geqslant f(a_t^z)$，$\forall z$，$1\leqslant z\leqslant m_c$，对于$a_t^z\in\{a_t^1, \cdots a_t^{m_c}\}$是分类值，$z$和$m_c$是分类属性$t$的类别数量. $f(a_t^z)=|\{x_{it}=a_t^z\mid p_{ij}=1\}|$是属性值$a_t^z$的频率计数，对于分类属性，当$p=q$时，$\delta(p, q)=0$，$p\neq q$时，$\delta(p, q)=1$.

2 基于Spark的大数据K-prototypes算法

本文提出的处理混合大规模数据的方法由Spark框架下的K-prototypes算法并行化实现. 首先，输入数据集被分成m个块，然后每个块在map阶段被独立处理，从每个块中提取中间中心，之后reduce阶段处理该组的中间中心以便生成最终聚类中心，这一步被称为重新聚集技术.

为了定义并行实现，有必要定义应用于每个块上的算法以及应用于该组中间中心的算法. 对于map和reduce这2个阶段使用K-prototypes算法，对于每个块执行K-prototypes算法并提取k个中心. 因此，如果有m块，K-prototypes算法在每个块上都会有一组k×m中心作为中间集合.

为了获得好的质量，记录每个提取中心的分配数据点数量，即从每个块、K中心和分配给每个中心的数据点数量中提取. 分配给每个聚类中心的数据点数表示该中心的重要性. 因此，必须扩展k-prototypes算法，在对中间中心集合进行聚类时考虑加权数据点. 为了考虑加权数据点必须改变中心更新(公式(3)和公式(4))，将w_i作为数据点x_i的权重，则最终簇的中心使用以下方程计算数值和分类属性.

$$c_{jr}=\frac{\sum_{i=1}^{|c_j|}x_{ir}*w_i}{|c_j|} \tag{5}$$

$$\begin{cases}c_{jt}=a_t^h\\ f(a_t^h)*w_i\geqslant f(a_t^z)*w_i\end{cases} \tag{6}$$

其中，$\forall z$，$1\leqslant z\leqslant m_c$.

通过 Spark 框架并行实现 K-prototypes 算法非常简单. 作为输入，从 HDFS 接收输入数据集 X 的路径，同时还处理组块数 m，组数 k. 首先，用 m 块组成的输入数据集 X 创建一个 RDD 对象，因此使用 Spark 框架下的 textfile(.)操作. 之后，每个 map 任务选择一大块数据集，在该块数据集上执行 K-prototypes 算法，并发出提取的中间中心及其权重作为输出. 在 Spark 实现中，使用 MapPartition(.)转换，它分别在 RDD 的每个块上运行 k-prototypes 算法. 当 map 阶段完成后，一组中间加权中心作为 map 阶段的输出，并且这组中心被输入到单个 reduce 阶段，reduce 阶段采用中间中心集合及其权重，再次执行 k-prototypes 算法并返回最终中心作为输出. 为了简化这个想法的实现，使用 Spark 的 ReduceByKey(.)转换，当生成最终聚类中心后，将每个数据点分配到最近的聚类中心.

设 X-RDD 为输入数据集的 RDD 对象，X_i 是与 map 任务 i 相关联的分区，$C^w=\{C_1^w,\cdots,C_m^w\}$ 为加权中间中心的集合，其中 C_i^w 是从组块 i 中提取的一组加权中间中心，C^f 为最后聚类中心的集合. 算法 1 描述了本文方法的伪代码以及 Spark 中使用函数的具体步骤.

算法 1　本文方法伪代码

输入：数据集 $\boldsymbol{X}$，map 任务数 m，聚类数量 k；
输出：最终聚类中心 C^f
begin
　textfile ($\boldsymbol{X}$, m)→X-RDD
　foreach $X_i\in$ X-RDD do
　　X-RDD. MapPartition(X_i)
　　在 X_i 上运行 K-prototypes 算法以获得一组 k 加权中心 C_i^w.
　　输出<1/C_i^w>作为 map 任务的输出
　ReduceByKey(C^w)
　在 C^w 上运行 K-prototypes 算法得到一组最终中心 C^f
　输出<1/C^f>作为 reduce 阶段的输出
end

算法 2 中给出了 K-prototypes 算法的具体步骤.

算法 2　K-prototypes 算法步骤

输入：数据集 $\boldsymbol{X}$，聚类数量 k；
输出：最终聚类中心

begin
　1. 从 $\boldsymbol{X}$ 中随机选择 k 个初始聚类中心.
　2. 将 $\boldsymbol{X}$ 中的每个数据点分配给最近的中使用等式(2)计算距离.
　3. 使用等式(3)和(4)更新聚类中心.
　　4. 如果新的聚类中心和以前的聚类中心相同，则终止；否则，返回
end

3　实验与结果分析

3.1　实验数据集及评价指标

为了评估 KP-S 方法的效率，本文对模拟和实际数据集进行了实验. 首先，本文实验比较了所提出的

方法、K-prototypes方法和基于MapReduce的K-prototypes(KP-MR)方法[13]的性能. 其次，通过分析所提出方法的可扩展性来评估Spark框架的性能.

实验使用Apache Spark版本1.6.2，Apache Hadoop 1.6.0和Ubuntu 14.04实现. 实验使用的数据集分为模拟数据集和真实数据集：

模拟数据集：生成4组混合大数据集. 数据集范围从500万～1亿个数据点. 每个数据点使用5个数字和5个分类属性进行描述. 数值是用高斯分布生成的，平均值为350，标准差为100，用数据生成器生成分类值. 为了简化模拟数据集的名称，使用符号Sim1,Sim2,Sim3和Sim4分别表示包含500万，1 000万，5 000万和1亿个数据点的模拟数据集.

KDD Cup数据集(KDD)：是一个真实的数据集，KDD数据集包含大约500万个数据点. 每个数据点都使用33个数字和3个分类属性进行描述.

Poker数据集(Poker)：是一个真实的数据集，包含从52张牌的标准牌组中抽取扑克牌的例子，扑克数据集包含大约100万个数据点，每个数据点使用5个数字属性和5个分类来描述.

为了评估所提出方法的质量，使用平方和误差(SSE)，旨在测量每个数据点与数据点所属聚类中心之间的平方误差，可定义为

$$SSE = \sum_{i=1}^{n}\sum_{j=1}^{k} d(c_j, x_i) \tag{7}$$

其中，x_i 是数据点，c_j 是聚类中心.

为了评估所提出方法处理大规模数据集的能力，在实验中使用加速指标，目的是测量设计的并行方法的扩展能力，计算得

$$Speedup = \frac{T_1}{T_m} \tag{8}$$

其中，T_1 是一台机器上处理数据的运行时间，T_m 是处理所使用聚类中 m 台机器上数据的运行时间.

3.2 实验结果分析

在本节中实验比较了本文方法与现有方法的性能，对模拟数据集和真实数据集进行3种算法的性能对比，并在其上应用了S-KP,KP-MR和K-prototypes方法. 表1是对4个模拟数据集的实验结果，k 值取50，表2和表3是对2个真实数据集的实验结果，k 值取10,50和100.

分析表1、表2和表3中数据可知，本文方法在运行时间上总是比K-prototypes方法和MapReduce上的K-prototypes方法快. 表3中Poker数据集实验数据中，聚类数量为100时，K-prototypes和MapReduce上的K-prototypes方法比较，本文方法分别可将运行时间减少96.47%和85.72%.

在所有的数据集中，超过95%的运行时间花在map阶段，这表明S-KP方法是真正在内存中执行的，此外，本文方法收敛到和K-prototypes几乎相同的SSE结果，但用时更少.

表1 模拟数据集实验对比结果

数据集	k	方 法	运行时间/s	SSE
Sim1	50	K-prototypes	1 268.24	315.88
		KP-MR	212.57	315.88
		S-KP	25.98	319.44
Sim2	50	K-prototypes	2 365.10	631.74
		KP-MR	394.11	631.74
		S-KP	54.17	637.58
Sim3	50	K-prototypes	13 698.84	3 158.66
		KP-MR	2 285.65	3 158.66
		S-KP	203.47	3174.06
Sim4	50	K-prototypes	26 874.12	6 124.12
		KP-MR	4 487.01	6 124.12
		S-KP	369.45	6 360.88

表 2　KDD 数据集实验对比结果

k	方　法	运行时间/s	*SSE*
10	K-prototypes	984.95	663.46
	KP-MR	246.37	663.46
	S-KP	23.31	992.23
50	K-prototypes	1 126.63	540.27
	KP-MR	280.74	540.27
	S-KP	25.46	752.35
100	K-prototypes	1 256.14	472.9
	KP-MR	311.96	472.9
	S-KP	27.13	700.35

表 3　Poker 数据集实验对比结果

k	方　法	运行时间/s	*SSE*
10	K-prototypes	127.9	1.39
	KP-MR	31.97	1.39
	S-KP	4.17	1.44
50	K-prototypes	146.65	1.08
	KP-MR	36.66	1.08
	S-KP	5.75	1.1
100	K-prototypes	170.07	0.97
	KP-MR	42.51	0.97
	S-KP	6.11	0.99

为了研究加速值，本文使用 Sim4 数据集进行实验，k 取值 100. 首先使用一台机器执行 S-KP 方法，然后添加额外的机器. 图 1 显示了 Sim4 数据集的加速结果，所提出的方法随着机器数量的增加而呈现线性加速，因为 Spark 框架具有线性加速并且每个块可以独立处理.

为了研究数据集大小对本文算法的影响，当增加数据集大小时，评估本文方法的可扩展性. 实验数据集是模拟数据集 Sim1-Sim4，k 取值 100，结果如图 2 所示.

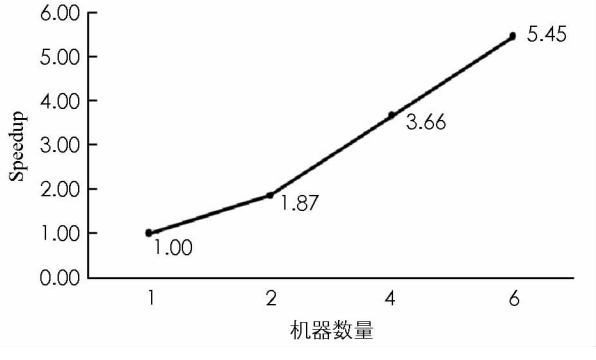

图 1　S-KP 的 Speedup 与机器数量的关系

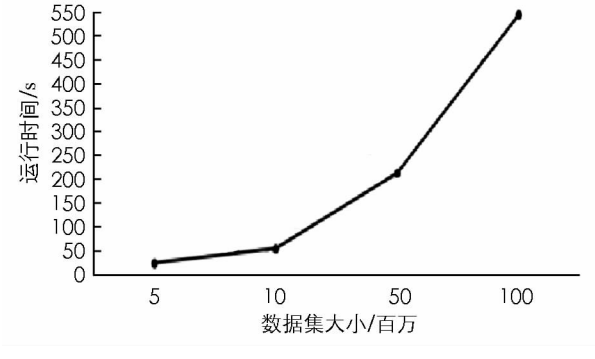

图 2　S-KP 运行时间与数据集大小的关系

从图 2 中可以看出，当数据集大小增加时，运行时间呈线性增长，如 KP-MR 方法[13]在 Sim4 数据集上花费超过 1 h，而本文方法在不到 10 min 的时间内处理了数据集.

4　结　　论

本文提出了一种新的基于 Spark 框架的 K-prototypes 聚类方法，在处理大规模混合数据时，K-prototypes 算法有 2 个主要问题：运行时间和内存消耗. Apache Spark 的引入为 k-prototypes 算法的并行化提供了一个简单、透明和高效的环境. 实验结果表明，本文方法是可扩展的，可以提高现有 K-prototypes 方法的效率. 未来的工作是通过使用降维技术，来处理具有大量特征的大规模混合数据.

参考文献:

[1] 车宝真,蔚承建,万夕里,等. 基于Spark平台的心电大数据分析处理[J]. 计算机工程与设计,2018,39(1):108-114.

[2] 郑子伟,郑建秋. 适用于大数据的遗传优化算法研究[J]. 西南师范大学学报(自然科学版),2016,41(12):107-112.

[3] 张顺龙,库 涛,周 浩. 针对多聚类中心大数据集的加速K-means聚类算法[J]. 计算机应用研究,2016,33(2):413-416.

[4] 海 沫. 大数据聚类算法综述[J]. 计算机科学,2016,43(S1):380-383.

[5] AYECH M W, ZIOU D. Segmentation of Terahertz Imaging Using K-Means Clustering Based on Ranked Set Sampling[J]. Expert Systems with Applications, 2015, 42(6): 2959-2974.

[6] YIN S, HUANG Z H. Performance Monitoring for Vehicle Suspension System Via Fuzzy Positivistic C-Means Clustering Based on Accelerometer Measurements[J]. ASME Transactions on Mechatronics, 2015, 20(5): 2613-2620.

[7] 何育朋. 混合的大规模数据库中数值型数据聚类算法研究[J]. 微电子学与计算机,2017,34(2):119-122,127.

[8] JANG H J, KIM B, KIM J, et al. Correction: An Efficient Grid-Based K-Prototypes Algorithm for Sustainable Decision Making Using Spatial Objects[J]. Sustainability, 2018, 10(8): 1-20.

[9] JI J C, BAI T, ZHOU C G, et al. An Improved K-Prototypes Clustering Algorithm for Mixed Numeric and Categorical Data[J]. Neurocomputing, 2013, 120: 590-596.

[10] LUDWIG S A. MapReduce-Based Fuzzy C-Means Clustering Algorithm: Implementation and Scalability[J]. International Journal of Machine Learning and Cybernetics, 2015, 6(6): 923-934.

[11] SHAHRIVARI S, JALILI S. Single-Pass and Linear-Time K-Means Clustering Based on MapReduce[J]. Information Systems, 2016, 60: 1-12.

[12] KIM Y, SHIM K, KIM M S, et al. DBCURE-MR: An Efficient Density-Based Clustering Algorithm for Large Data Using MapReduce[J]. Information Systems, 2014, 42: 15-35.

[13] BEN HAJ KACEM M A, BEN N'CIR C E, ESSOUSSI N. MapReduce-Based K-Prototypes Clustering Method for Big Data[C]//2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA). Paris: Campus des Cordeliers, 2015.

K-prototypes Clustering Algorithm Based on Spark Framework for Big Data

GONG Jing

School of Data Science, Tongren University, Tongren Guizhou 554300, China

Abstract: Big data has a large amount of data and mixed types of attributes. The disadvantages of the current MapReduce-based K-prototypes parallel large-scale hybrid data plan are the limitations of time and memory, making these solutions unsuitable for processing big data. To solve this problem, a new Spark-based K-prototypes clustering method has been proposed in this paper. In this method, the re-aggregation technique and Spark's memory operations have been used to build large-scale mixed data groups. Experiments on simulated and actual datasets show that this method is feasible and improves the efficiency of the existing K-prototypes method.

Key words: big data; mixed data; K-prototypes; spark framework

责任编辑 夏 娟