基于 Spark 平台的 K-means 算法的设计与优化

王义武[1],杨余旺[1],于天鹏[2],沈兴鑫[1],李猛坤[3]
(1. 南京理工大学 计算机科学与工程学院,江苏 南京 210000;
2. 304 兵器厂,山西 长治 046000;
3. 清华大学 经管学院,北京 100000)

摘 要: 聚类中心需要手动设置是 K-means 算法最大的问题,而通常情况是并不能确定现实中数据的分类情况。为了解决这一问题,提出了一种新的 OCC K-means 算法。不同于传统算法以随机选择的方式产生聚类中心,该算法进行必要的预处理,利用 UPGMA 和最大最小距离算法对数据点进行筛选,得到可以反映数据分布特征的点,并作为初始的聚类中心,以提高聚类的精度。从两次的实验结果可以对比出,在不同的数据集上,改进算法在衡量聚类效果的准确率、召回率、F-测量值上的表现要优于传统 K-means 算法。这是因为 OCC 算法选择的中心点来自于不同的且数据密集的区域,并在筛选的过程中排除了噪声数据、边缘数据对实验的干扰;同时为了契合大数据发展潮流,使用 Scala 语言在 Spark 平台进行了并行化实现,提高了算法处理海量数据的能力,并通过实验指标验证了算法具有良好的并行化能力。

关键词: 聚类;聚类中心;K-means;最大最小距离算法;非加权组平均法
中图分类号: TP301 **文献标识码:** A **文章编号:** 1673-629X(2019)03-0072-05
doi:10.3969/j.issn.1673-629X.2019.03.015

Design and Optimization of K-means Algorithm Based on Spark Platform

WANG Yi-wu[1],YANG Yu-wang[1],YU Tian-peng[2],SHEN Xing-xin[1],LI Meng-kun[3]
(1. School of Computer Science and Engineering,Nanjing University of Science and Technology,
Nanjing 210000,China;
2. 304Weapon Factory,Changzhi 046000,China;
3. School of Economics and Management,Tsinghua University,Beijing 100000,China)

Abstract: The clustering center needs to be set manually is the biggest problem of K-means algorithm,and it is usually impossible to determine the classification of data in reality. In order to solve the problem,we propose a new OCC K-means algorithm. Different from the traditional algorithm,which generates the clustering center in the way of random selection,this algorithm carries out necessary preprocessing,and uses UPGMA and maximum and minimum distance algorithm to screen data points for the ones that can reflect data distribution characteristics as the initial clustering center,so as to improve the accuracy of clustering. From the two experimental results,it can be found that in different data sets,the improved algorithm is better in the measurement of clustering accuracy,recall,F-measurement than the traditional K-means algorithm. This is because the center point selected by OCC algorithm comes from different and data-intensive areas,and noise data and edge data interference to the experiment are excluded in the process of screening. At the same time,in order to conform to the trend of big data development,the parallelization implementation is carried out on Spark platform with Scala language,which improves the ability of the algorithm to deal with massive data,and the better parallelization of the algorithm is verified by experimental indexes.

Key words: clustering;clustering center;K-means;maximum and minimum distance algorithm;unweighted pair group method with arithmetic mean

收稿日期: 2018-03-19 **修回日期:** 2018-07-31 **网络出版时间:** 2018-12-19
基金项目: 国家自然科学基金(61640020);江苏省农业自主创新项目(CX(13)3054、CX(16)1006);江苏省重点研发计划(BE2016368-1);江苏省科技重点及面上项目(SBE2018310371);弹总装线***技术研究(JCKY2017***);Postgraduate Research&Practice Innovation Program of Jiangsu Province(SJCX17_0107);北京市教育委员会科技计划面上项目(KM201510028019)
作者简介: 王义武(1992-),男,硕士研究生,研究方向为数据分析;杨余旺,教授,研究方向为大数据系统、计算机网络、网络编码。
网络出版地址: http://kns.cnki.net/kcms/detail/61.1450.TP.20181219.1510.008.html

0 引　言

从算法复杂程度看,K-means 算法简洁、高效,其他很多算法难以相比,因此其应用愈加广泛,但它也存在固有问题,如聚类中心需人为主观设置,但只有了解数据分布才能掌握相关信息,这与现实相矛盾; 聚类效果和中心点选择有关,易陷入局部最优解[1],中心点的不同选择,可能造成结果大不相同; 传统 K-means 需要依序逐个计算数据点之间的距离[2],当数据规模很大时在计算上耗费了大量时间。针对这些固有问题,不断有人提出优化方法,使得 K-means 算法愈加成熟、多样。

随着分布式系统的出现与完善,传统算法利用分布式计算的优点在聚类效果上得到了不断的优化和提高。文献[3]实现了 Hadoop 版本的并行处理,提高了算法的吞吐量; 文献[4]在 Hive 平台下实现了 K-means 的分布式计算且运用简单的类 SQL 语言即可以实现数据查询; 文献[5]利用 Canopy 优化 K-means 算法,并在 MapRduce 框架下实现。在上述研究的基础上,文中提出一种新的 OCC K-means 算法。不同于传统算法以随机选择的方式产生聚类中心,该算法进行必要的预处理,利用 UPGMA 和最大最小距离算法对数据点进行筛选,得到可以反映数据分布特征的点,并作为初始的聚类中心,以提高聚类精度。

1 相关工作

1.1 Spark 分布式框架

Spark 作为一种正兴起的计算框架,建立在弹性分布式数据集基础之上,具有以下特点:

(1) 数据分布式存储在多个节点上;

(2) 高度抽象化的 RDD 概念,这也是 Spark 与 Hadoop、Hive 最大的不同,RDD 是一切计算的基础,这也使得 Spark 迭代计算、并行计算的能力明显好于其他现存的分布式框架;

(3) 不同于 Hadoop 频繁地在 HDFS 进行 I/O 操作,Spark 计算时将数据写入内存,有效降低了计算时间,同时当内存不够时,它可以将数据写到磁盘上;

(4) 高度的容错性,Spark 抽象出 RDD 并在计算时只记录了各个 RDD 之间的关联,没有真正地产生相关数据,仅在 Action 动作时才真正提交作业。

Spark 运行模式多样,部署在单机上时可以选用本地模式(local)或者伪分布模式(local cluster); 当部署在集群上时可以选用 Standalone 模式、Mesos 模式或者 Hadoop YANR 模式。这里具体的介绍 Standalone 模式[6]。

1.2 基本概念

定义 1: 点 P_i 和 P_j 之间采用欧氏距离,定义为:

$$\mathrm{Dist}(P_i,P_j)=\mathrm{sqrt}\left(\sum_{l=1}^{n}(P_{il}-P_{jl})^2\right) \tag{1}$$

其中,P_i ,P_j 为 n 维数据。

定义 2: 簇 C 的中心定义为 $C=(C_1,C_2,\cdots,C_N)$,数据点第 n 维值为:

$$C_n=\frac{\sum_{i=1}^{m}X_{in}}{m} \tag{2}$$

其中,m 为数据个数; X_i 为第 i 个数据; n 为维度。

定义 3: d 为数据 j 距集合 i 中所有的数据的最小距离。

$$d=\mathrm{Min}(\mathrm{Dist}(U_{ik}-U_j)) \tag{3}$$

其中,U_{ik} 为数据集 i 中第 k 个数据; U_j 标记数据集 j。

定义 4: 由定义 3 进一步得出最大最小距离为:

$$D_{\max}=\mathrm{Max}(d_{ij}^{*}) \tag{4}$$

其中,d_{ij}^{*} 为数据集 i 中各元素与数据集 j 最小距离的集合。

定义 5: 最大最小距离算法[7-8]中用于选出分布较远的数据点。

$$\mathrm{Di}_{\max}>\theta\|v_1-v_2\| \tag{5}$$

其中,v_1 ,v_2 为数据集 i ,j 间距离最小的两个数据点; θ 为算法参数,值取(0,1)之间。

定义 6: 算法的收敛判定标准-误差平方和准则函数。

$$J_c=\sum_{i=1}^{k}\sum_{p\in C_i}(P-M_i)^2 \tag{6}$$

其中,M_i 是簇 C_i 的中心点; p 是簇 C_i 的数据点元素; k 为簇的个数。在给定数据集时,J_c 表示将数据集划分为 k 个簇的总体误差。从式 6 可以看出,M_i 决定了 J_c 取值大小,J_c 值越大表示数据划归的误差越大。因此在聚类过程中应致力于 J_c 最小化,从而使聚类最优化[9]。

定义 7: 准确率和召回率。

$$P(i,j)=\mathrm{precision}(i,j)=N_{ij}/N_i \tag{7}$$

$$R(i,j)=\mathrm{recall}(i,j)=N_{ij}/N_j \tag{8}$$

其中,N_{ij} 表示属于类 i ,但被划归到类 j 中的数据数; N_i 表示类 i 中全部数据点个数; N_j 表示类 j 中全部数据点的个数。由式 7 和式 8 可得 F-measure:

$$F(i)=2P*R/(P+R) \tag{9}$$

其中,$F(i)$ 结合准确率和召回率,表示相对准确率。

定义 8: 加速比(sizeup)。

$$S=\frac{T_s}{T_n} \tag{10}$$

其中,T_s 为一个 Worker 算法完成时间; T_n 为 n 个 Worker 算法完成时间。

定义9: 扩展比(scaleup) 。

$$E = \frac{S}{n} \tag{11}$$

其中, S 为加速比; n 为 Worker 节点的个数。

2 OCC K-means 并行算法

2.1 算法思想

传统 K-means 算法对聚类中心过于敏感,这也导致聚类效果上的显著不同,因此文中提出使用 UPGMA 算法[10] 筛选出可以反映数据集整体分布的候选中心,同时被选择的数据点应该是相距比较远的,防止过于集中[11-12] ,以避免同一个聚类因为参数设置的原因选出了两个聚类中心,这样 K-means 算法就获得了一个较好的输入,并且在整个执行过程中 UPGMA 算法和 K-means 算法两个阶段都是并行的。

UPGMA 阶段的执行过程如下:

(1) 读入 HDFS 上将要聚类的数据,初始情况下每一个数据点都看作一个类,并通过 Map 将其转化为 Vector;

(2) Worker 节点的各个数据计算距其他数据点的最小欧氏距离,通过 Reduce 操作将距离最近的两个数据点合并,计算它们的中心作为新的数据点(同时记录这个数据点代表的类所拥有的数据的数目) 参与聚类;

(3) 若类内的数据大于数据点总数的 $m\%$,则将此数据点加入到候选列表,通过 Filter 操作过滤掉已存在于候选列表中的类中数据; 否则重复步骤2;

(4) 当无归属的数据点的数目小于总数的 $m\%$ 时,算法停止,输出候选列表作为最大最小距离算法的输入数据; 否则一直重复上述过程。

K-means 阶段的执行过程如下:

(1) 将最大最小距离算法的输出作为 K-means 算法的输入;

(2) Worker 查找距离自己最近的中心点;

(3) 将距离同一个初始中心最近的所有数据点通过 Iterator 对距离全局求和、记录数据个数,然后加入到同一个类中,更新此类的中心;

(4) 若算法收敛则停止,否则返回步骤2。

2.2 基于 Spark 的 OCC K-means 算法的并行化

2.2.1 UPGMA 算法的并行化

使用 UPGMA 算法主要目的是为了了解数据的分布,找到数据的密集区域,从中选出代表这一区域的数据点。

改进的 UPGMA 算法的并行化过程如下:

(1) 算法输入: 待聚类的数据集, m , p , θ; 算法输出: 初始候选中心集合。

(2) 将单个数据作为独立的类。

(3) 计算类间彼此距离,合并相距最近的两个类,同时判断剩下数据的总数是否大于等于总数的 $m\%$,若不大于等于则转步骤4。

(4) $i=0, t=0, t=t+1$, for $j=i+1...$ maxcluster do

(5) 当子类 i 和子类 j 内部的数据个数少于等于总数的 $m\%$,且两个类数据元素之和大于总数的 $m\%$ 时,计算两个类之间距离,并加入到距离矩阵中。

(6) 合并距离矩阵中最近的类,并且加入到序列 Q ,转至步骤3。

(7) 选取序列 Q 的前 $p\%$,计算中心位置,即为初始候选中心。

$m\%$ 为类中的数据点占数据总数的百分比,当合并后的类中数据点的个数满足这一比例时才会将其加入到序列 Q; 最后选取序列 Q 的前 $q\%$ 。

2.2.2 K-means 算法的并行化

经过上面两个过程,K-means 算法获得了一个较好的输入,这里主要说明 K-means 并行化的实现。

Map 操作:

输入: 待聚类数据集 S ,初始聚类中心数组 Centerlist

输出: 已被标记所属聚类的数据集合

(1) while(S! = null) ,计算 P (S 中的数据) 与 Centerlist 中第 i 个数据的距离,记为 D_{pi} ,加入集合 $\{D_{pi}\}$ ($i=1,2,\cdots$,Centerlist. length) ,在 S 中删除 P 。

(2) 找出 $\{D_{pi}\}$ ($i=1,2,\cdots$,Centerlist. length) 中最小的所对应的中心 D_{pi} ,将 P Map 为 (p,i)

Combine 操作:

相同 i 构成的集合(类) 对 Map 产生的数据进行分类、汇总

Reduce 操作:

(1) while(C. hasNext( )) , num = 0, double du [demesions] , $P=C.$ next( ) ,将数据点 P 的第 i 分量值加入 du [ ] 中, num ++

(2) 输出类中心 du [ ] /num

每一次 reduce 操作都会更新一次聚类中心,当前后两次中心的值不发生变化或者变化满足停止条件时说明已经收敛,此时算法停止。

3 实验结果与分析

3.1 实验环境、测试数据集及评价指标

实验是在 Spark 的 Standalone 模式下进行的,Spark 采用1.4.0版本,Hadoop 采用2.6.0版本,Java jdk 为1.8,Master 和 Worker 节点的操作系统为 ubuntu 14.04 LTSk,其中 Worker 节点的个数为5。

3.2　实验过程

这里进行了两组实验,使用传统 K-means 和 OCC K-means 对三个数据集进行处理。实验中采用的数据集为 Iris,Seeds,Dum,OCC K-means 算法中的参数依次设置为:(0.8,0.8,0.5)、(0.5,0.8,0.5)、(0.5,0.8,0.5),两组实验结果如表 1 所示。

表 1　实验结果

实验数集	评价标准	K-means	OCCK-means
Iris	准确率	0.853 42	0.886 60
	召回率	0.807 56	0.895 32
	F-measure	0.829 86	0.890 94
Seeds	准确率	0.866 36	0.892 27
	召回率	0.855 53	0.891 34
	F-measure	0.860 91	0.891 80
Dum	准确率	0.422 25	0.564 41
	召回率	0.536 43	0.623 63
	F-measure	0.472 54	0.592 54

效果对比如图 1 ~ 3 所示; OCC K-means 算法加速比和扩展比分别如图 4 和图 5 所示。

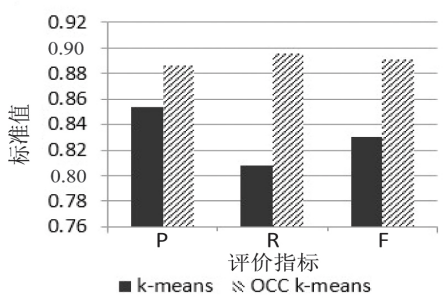

图 1　Iris 数据集上的效果对比

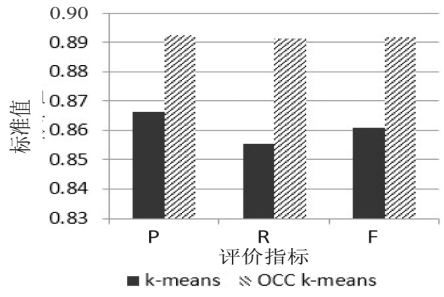

图 2　Seeds 数据集上的效果对比

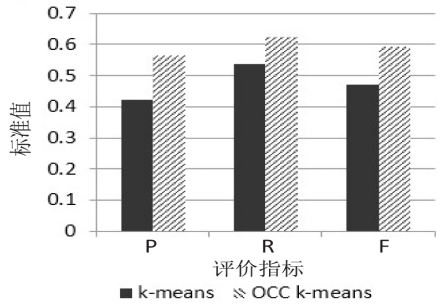

图 3　Dum 数据集上的效果对比

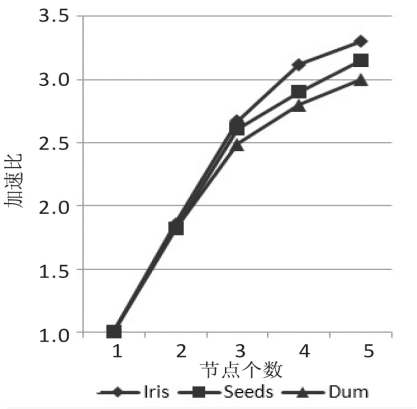

图 4　OCC K-means 算法加速比

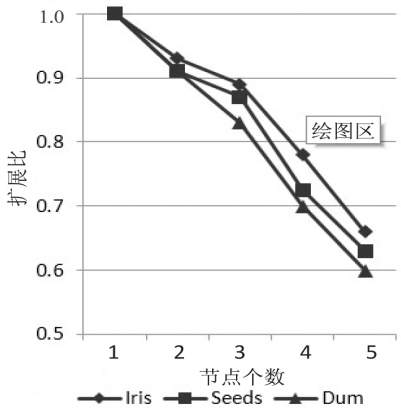

图 5　OCC K-means 算法扩展比

可以发现在 Iris 数据集上,OCC K-means 算法相比传统 K-means 算法在准确率上提高了 3.33%,召回率提高了 8.78%,*F*-measure 提高了 6.11%;在 Seeds 的数据集上,准确率提高了 2.59%,召回率提高了 3.58%,*F*-measure 提高了 3.09%;在 Dum 数据集上,准确率提高了 14.22%,召回率提高了 8.72%,*F*-measure 提高了 12.00%,可见改进后的算法在聚类效果上得到了优化[13]。原因在于 OCC K-means 算法使用 UPGMA 算法和最大最小距离算法对聚类中心进行了选择优化,这些中心点来自数据密集区域可以很好地代表数据的分布,同时在一定程度上将噪声数据和边缘数据的影响尽可能降低,使得算法体现出更好的准确性和鲁棒性[14-15]。

同时对加速比和扩展比的分析可知,当 Worker 的节点小于 3 时,算法的加速比近似于线性上升,但随着节点数增大而增幅出现下降,这是因为虽然硬件资源成倍增加,但它们并没有全部得到使用,反而集群初始化时间,资源调度的代价,节点间通信代价等会增加,这些都会影响算法的并行效果。

4　结束语

从两组实验结果可以得出,基于 Spark 实现的 OCC K-means 算法,在准确率、召回率、*F*-measure 上

的表现都优于传统 K-means，并且在 Iris、Seeds、Dum 不同数据集上均取得了较好的结果，说明 K-means 算法在聚类中心选择上得到了一定程度的优化，拥有更好的刻画数据分布特征的能力。同时当工作节点数量小于一定值时，加速比、扩展比可出现理想的变化，说明算法具有较好的并行化能力，但是当工作节点过多或者实验数据集过小时，算法的并行化效果就会受到影响，有待进一步完善。

参考文献：

[1] 刘 辉. 某些条件下的极大极小系统的全局最优解 [D]. 石家庄：河北师范大学，2017.

[2] 张忠林，曹志宇，李元韬. 基于加权欧式距离的 k_means 算法研究 [J]. 郑州大学学报：工学版，2010，31(1)：89-92.

[3] 贾瑞玉，管玉勇，李亚龙. 基于 MapReduce 模型的并行遗传 k-means 聚类算法 [J]. 计算机工程与设计，2014，35(2)：657-660.

[4] 冯晓云，陆建峰. 基于 Hive 的分布式 K_means 算法设计与研究 [J]. 计算机光盘软件与应用，2013(21)：62-64.

[5] 刘宝龙，苏 金. 双 MapReduce 改进的 Canopy-Kmeans 算法 [J]. 西安工业大学学报，2016，36(9)：730-737.

[6] 夏俊鸾，程 浩，邵赛赛，等. Spark 大数据处理技术 [M]. 北京：电子工业出版社，2015：67-68.

[7] 侯 玥. 基于最大最小距离聚类算法的改进多重心法选址研究 [D]. 大连：辽宁师范大学，2015.

[8] 成卫青，卢艳红. 一种基于最大最小距离和 SSE 的自适应聚类算法 [J]. 南京邮电大学学报：自然科学版，2015，35(2)：102-107.

[9] ABED H，ZAOUI L. Partitioning an image database by K_means algorithm [J]. Journal of Applied Sciences，2011，11(1)：16-25.

[10] 季 赛，谭 畅. 基于 UPGMA 聚类无线传感网络的簇头选择方法 [J]. 武汉理工大学学报，2010，32(16)：139-142.

[11] 翟东海，鱼 江，高 飞，等. 最大距离法选取初始簇中心的 k-means 文本聚类算法的研究 [J]. 计算机应用研究，2014，31(3)：713-715.

[12] 于彦伟，王 沁，邝 俊，等. 一种基于密度的空间数据流在线聚类算法 [J]. 自动化学报，2012，38(6)：1051-1059.

[13] SUN Qiao. An efficient distributed database clustering algorithm for big data processing [C]//Proceedings of 2017 3rd international conference on computational systems and communications. [s. l.]：[s. n.]，2017.

[14] GUHA S，RASTOGI R，SHIM K，et al. CURE：an efficient clustering algorithm for large databases [C]//ACM SIGMOD international conference on management of data. [s. l.]：ACM，1998：73-84.

[15] CUI Chunchun. Parallel CSA-FCM clustering algorithm based on MapReduce [C]//Proceedings of 2017 international conference on sports，arts，education and management engineering. [s. l.]：[s. n.]，2017.

(上接第 71 页)

之间的相似性并使用 CF 预测它们的推荐。以前相关的方法在基于内容的算法中使用了朴素贝叶斯等文本分类器，而文中算法进一步测试并与 Pure CF 和 SVD 进行了比较。评估后生成的 MAE 值提供了成功的比较。虽然混合内容推荐可产生更好的 MAE 值，并将数据集的稀疏性提高了 1% ~2%，但使用较大的数据集进行测试时，结果可能会有所不同。

参考文献：

[1] 李 川. 实时个性化推荐系统的设计与实现 [D]. 北京：北京邮电大学，2015.

[2] 马晓迪，宣 琦，张 哲，等. 协同过滤推荐算法在豆瓣网络数据上的研究 [J]. 计算机系统应用，2014，23(8)：18-24.

[3] HAN J，ISHII M，MAKINO H. A Hadoop performance model for multi-rack clusters [C]//5th international conference on computer science and information technology. Amman，Jordan：IEEE，2013：265-274.

[4] 黄宜华. 深入理解大数据：大数据处理与编程实践 [M]. 北京：机械工业出版社，2013.

[5] WU Xindong，ZHU Xingquan，WU Gongqing，et al. Data mining with big data [J]. IEEE Transactions on Knowledge and Data Engineering，2014，26(1)：97-107.

[6] MANYIKA J，CHUI M，BROWN B，et al. Big data：the next frontier for innovation，competition，and productivity [R]. [s. l.]：[s. n.]，2011.

[7] 胡 涛，周 兵，郑明辉，等. 基于 Hadoop 的移动云存储系统研究与实现 [J]. 华中科技大学学报：自然科学版，2013，41：181-183.

[8] 潘天鸣. 基于 Hadoop 平台的决策树算法并行化研究 [D]. 上海：华东师范大学，2012.

[9] 王全民，苗 雨，何 明，等. 基于矩阵分解的协同过滤算法的并行化研究 [J]. 计算机技术与发展，2015，25(2)：55-59.

[10] 杨志伟. 基于 Spark 平台推荐系统研究 [D]. 合肥：中国科学技术大学，2015.

[11] 艾聪聪. 推荐系统中多样性和新颖性算法研究 [D]. 长沙：湖南大学，2014.

[12] CHEN Y，HARPER F M，KONSTAN J，et al. Social comparisons and contributions to online communities：a field experiment on movie lens [J]. American Economic Review，2010(4)：1358-1398.

[13] 刘建国，周 涛，郭 强，等. 个性化推荐系统评价方法综述 [J]. 复杂系统与复杂性科学，2009，6(3)：1-10.

[14] 马建威，徐 浩，陈洪辉. 信息推荐系统中的朋友关系预测算法设计 [J]. 国防科技大学学报，2013，35(1)：163-168.