软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2015,26(11):2820−2835 [doi: 10.13328/j.cnki.jos.004902]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

大数据的密度统计合并算法*

刘贝贝[1], 马儒宁[1], 丁军娣[2]

[1](南京航空航天大学 理学院,江苏 南京 211100)

[2](南京理工大学 计算机科学与技术学院,江苏 南京 210094)

通讯作者: 马儒宁, E-mail: mrning@nuaa.edu.cn, http://www.nuaa.edu.cn

摘 要: 针对处理大数据时传统聚类算法失效或效果不理想的问题,提出了一种大数据的密度统计合并算法(density-based statistical merging algorithm for large data sets,简称 DSML).该算法将数据点的每个特征看作一组独立随机变量,并根据独立有限差分不等式获得统计合并判定准则.首先,使用统计合并判定准则对 Leaders 算法做出改进,获得代表点集;随后,结合代表点的密度和邻域信息,再次使用统计合并判定准则完成对整个数据集的聚类.理论分析和实验结果表明,DSML 算法具有近似线性的时间复杂度,能处理任意形状的数据集,且对噪声具有良好的鲁棒性,非常有利于处理大规模数据集.

关键词: 聚类;抽样;代表点;密度;大数据

中图法分类号: TP181

中文引用格式: 刘贝贝,马儒宁,丁军娣.大数据的密度统计合并算法.软件学报,2015,26(11):2820−2835. http://www.jos.org.cn/1000-9825/4902.htm

英文引用格式: Liu BB, Ma RN, Ding JD. Density-Based statistical merging algorithm for large data sets. Ruan Jian Xue Bao/Journal of Software, 2015,26(11):2820−2835 (in Chinese). http://www.jos.org.cn/1000-9825/4902.htm

Density-Based Statistical Merging Algorithm for Large Data Sets

LIU Bei-Bei[1], MA Ru-Ning[1], DING Jun-Di[2]

[1](College of Science, Nanjing University of Aeronautics and Astronautics, Nanjing 211100, China)

[2](School of Computer Science and Technology, Nanjing University of Science and Technology, Nanjing 210094, China)

Abstract: To tackle the failure of traditional clustering algorithms in dealing with large-scale data, the paper proposes a density-based statistical merging algorithm for large data sets (DSML). The algorithm takes each feature of data points as a set of independent random variable, and gets statistical merger criteria from the independent bounded difference inequality. To begin with, DSML improves Leaders algorithm by using the statistical merger criteria, and makes the improved algorithm as the sampling algorithm to obtain representative points. Secondly, combined with the density and the neighborhood information of representative points, the algorithm uses statistical merger criteria again to complete the clustering of the whole data set. Theoretical analysis and experimental results show that, DSML algorithm has nearly linear time complexity, can handle arbitrary data sets, and is insensitive to noise data. This fully proves the validity of DSML algorithm for large data sets.

Key words: clustering; sampling; leader; density; large data

聚类[1−3]作为一项重要的数据分析技术,是将给定的数据集划分成互不相交的非空子集的过程.每个非空子集代表一个类别,属于同一类别的数据点具有较高的相似性,而属于不同类别的数据点具有较大的差异性.聚类分析在模式识别、数据挖掘、信息处理、机器学习等领域都具有广泛的应用前景.目前已有多种聚类算法被开发应用.根据形成方式的不同,可以将其大致归纳为划分聚类算法和层次聚类算法[3].

* 基金项目: 国家自然科学基金(61103058, 61233011)
收稿时间: 2015-05-30; 修改时间: 2015-07-14, 2015-08-11; 定稿时间: 2015-08-26

划分聚类算法又可称为分割聚类算法.它首先根据给定的要构建的划分数目 k 选定每一类别的代表点,在遵循类内相似原则的情况下创建初始划分;然后,采用一种迭代的重定位技术尝试改变代表点的位置以优化评价准则函数值;最后,选取评价准则函数值最优时的划分为最佳分割.最具代表性且应用范围最广的划分聚类算法是 *K*-means 算法[4].该算法选取误差平方和函数作为评价准则函数,其对初始聚类中心和噪声点较为敏感,且面对非凸数据集时易陷入局部最优.随后,具有鲁棒性的 *K*-medoids 算法[5]应运而生,该算法用每个类别中与其他点距离之和最小的真实数据点来代表该簇,有效地消除了噪声点对聚类结果的影响.PAM 算法[6]是最早的 *K*-medoids 算法之一,其划分思想主要是针对数据集中成对的数据点,首先随机选取 k 个数据点作为代表对象,然后反复地用非代表对象代替代表对象,试图通过这种方式改进聚类质量,最终找到最优的聚类结果.但随着所处理数据集规模和簇的数目的增大,该算法的性能会随之变差.

层次聚类算法也可以称为树聚类算法.其主要思想是:对给定的数据集依照相似性矩阵进行层次分解,使得聚类结果可以由二叉树或系统树图描述.根据层次形成方式的不同,又可将该算法分为分裂式和凝聚式.分裂式层次聚类算法首先将数据集看作一类,然后依照类内最大相似性的原则将数据集逐渐细分,直到满足终止条件或每一个数据点构成一类时停止分裂.整体采用一种“自顶向下”的方式进行.由于对含有 N 个对象的数据集,该算法将其划分成两个子集有 $2^{N-1}-1$ 种可能,计算量是相对较大的,因此,分裂式层次聚类算法在实践中并不常用.MONA 和 DIANA 算法[7]都是分裂式层次聚类算法.凝聚式层次聚类算法的过程正好与分裂式相反,它首先将数据集的每个数据点看作一类,然后进行一系列的合并操作,直到满足终止条件或所有数据点归为一类时停止凝聚.整体采用一种“自底向上”的方式进行.常见的以距离作为类间相似性衡量准则的凝聚式层次聚类算法有 Single-link,Complete-link,Average-link 算法[6].三者的不同之处在于它们对类间距离的定义不同:Single-link 算法将两个不同类别内点与点的最小距离作为类间距离,而 Complete-link 和 Average-link 算法则分别将两个不同类别内点与点的最大距离和平均距离作为类间距离,这就导致合并过程中类别间的相似性不同.由于 Single-link 和 Complete-link 算法采用了距离运算的两个极端值,使得聚类结果无论是对噪声或是数据点的波动都很敏感;而 Average-link 算法采用的平均距离虽然不能合理地表示数据点的全局信息,但在一定程度上增加了算法对噪声的鲁棒性.

由于划分聚类算法(如 *K*-means 算法)大都只能聚类凸状或云团状数据集,而层次聚类算法(如 Single-link 算法)一般计算量大且对噪声点敏感,为弥补这些缺陷,提出了基于密度的聚类算法.该类方法从对象分布区域的密度着手,认为各目标簇都是由一些稠密对象所组成的,簇与簇之间被一些稀疏对象(如噪声)分割.算法的目的就是通过发现稠密对象,过滤稀疏对象,形成各种不同形状的聚类.最经典的基于密度的聚类算法为 DBSCAN 算法[8],该算法将每一数据点固定半径的邻域内所包含数据点的个数作为密度信息,然后利用密度阈值将数据集中的点分为稠密点和非稠密点,通过稠密点不断凝聚其邻域内的其他稠密点来形成聚类,当某类别中稠密点邻域内没有可凝聚的新稠密点时,该类别聚类结束.DBSCAN 算法在不需要人工输入聚类个数的情况下,不仅可以聚类任意形状的数据集,而且对噪声具有良好的鲁棒性.但该算法对涉及到的两个参数(邻域半径和邻域内数据点个数阈值)十分敏感,参数微小的变化都会导致聚类结果产生较大的差异.为了提高 DBSCAN 算法的聚类效率,在其基础上提出了多种改进算法,如 OPTICS 算法[9]、NBC 算法[10]等.OPTICS 算法并不直接将数据集中的对象分配给某一簇,而是通过额外存储每一对象的核心距离和可达距离获得一个有序的列表,随后,根据列表中的有效信息提取聚类.由于参数微小的变化对列表的排序影响不大,因此,OPTICS 算法对参数的选取并不敏感. NBC 算法的整体聚类思想与 DBSCAN 算法的相同,但在判断数据点是否为稠密点时,NBC 算法选用其 k 邻域与反 k 邻域中包含数据点个数的比值作为判断依据,将算法的参数减少为一个(最近邻个数 k).2014 年,在 Science 的一篇文章《Clustering by fast search and find of density peaks》[11]中,提出了一种基于密度的聚类算法.该算法将高于附近点的密度,且与更高密度的点距离相对较远的数据点定义为聚类中心.算法依据该定义,首先对每一数据点计算局部密度和与更高密度点的最小距离,然后,通过两者绘制的决策图来确定聚类中心.对剩余数据点,按照与它相距最近的更高密度点所属类别进行归类.该算法只需考虑点与点之间的距离信息,且对任意形状和含有噪声点的数据集都能很好地聚类,为基于密度的聚类算法提供了新的思路.

大多数传统聚类算法已经较为成功地解决了小规模数据的聚类问题,但是随着数据库技术和网络技术的迅猛发展,数据源开始不断地膨胀,人们获取的数据集规模逐渐增大,数据结构也变得日渐复杂.面对越来越多的具有类间相似和类内相异特征的大型数据集,传统聚类算法显得力不从心.表 1 中总结了几种经典传统聚类算法的时间和空间复杂度,可以看到,*K*-means 算法的时间复杂度在 n 远大于 K 的情况下,可看作与 n 成线性关系,这虽然满足了大规模数据聚类对算法复杂度的要求,但该算法包含 NP 难题,且迭代次数依赖于初始聚类中心,在实际处理大规模数据时极不方便;而其他几种算法的时间复杂度与 n 成非线性关系,随着数据集中包含对象的增多,时间复杂度会明显增大,这不仅影响到算法聚类的精确度,而且使算法在运行时间上也大大超出了现实的承受范围,因此它们不适合处理大规模数据.

Table 1 Time and space complexity of classic traditional clustering approaches

表 1 经典传统聚类算法的时间和空间复杂度

聚类算法	复杂度	
	时间	空间
K-means	$O(nKt)$	$O(K)$
K-medians	$O(n^2t)$	$O(n)$
K-mediods (PAM)	$O(K(n-K)^2)$	$O(K)$
Single/Complete link	$O(n^3)$	$O(n^2)$
DBSCAN	$O(n\log(n))$	$O(n)$

n:总对象个数; K:代表对象个数; t:迭代次数

处于大数据时代,如何解决大规模数据的聚类问题尤为重要.目前,研究人员从传统聚类算法失效的原因出发,以提高聚类算法的可扩展性和高效性为目的,提出了几种新颖的聚类思想作为解决方案[12]:

- 第 1 种是数据抽样:对原始数据集中具有代表性的子集进行聚类,然后在子集中找到各簇的代表点,将剩余数据点归到相似性最大的簇中.
- 第 2 种是并行处理:先将大数据集划分成小数据集,然后对每个小数据集同时进行聚类,将获得的聚类结果合并为最终聚类结果.
- 第 3 种是数据压缩:将原始数据集中的多个样本用其他占用内存较小的变量(如向量或矩阵)表示,从而降低算法所需内存.

这 3 种聚类思想均不受数据规模的限制,尤其适合处理大型数据集的聚类问题.

本文主要从数据抽样的角度对聚类算法进行研究,并在已有算法的基础上提出一种新的基于抽样的大数据聚类算法.第 1 节回顾几种经典的基于抽样的大数据聚类算法,并简单介绍新提出的聚类算法(density-based statistical merging algorithm for large data sets,简称 DSML).第 2 节详细描述 Leaders 算法和改进后的 Statistical Leaders 算法的聚类过程,同时,对 Statistical Leaders 算法进行实验分析.第 3 节给出完整的 DSML 算法的具体步骤、实现过程及时空复杂度,并将 DSML 算法与 Rough-DBSCAN 算法做出对比分析.第 4 节是对全文的总结.

1 研究基础

经典的基于抽样的大数据聚类算法包括 CLARA(CLARANS)算法、BIRCH 算法、CURE 算法等.

CLARA 算法[6]首先对原始数据集进行随机抽样,获得样本集;然后,利用 PAM 算法获得各簇的中心点.多次重复上述过程,最终得到最优聚类结果.CLARANS 算法[13]则在每一次循环过程中都进行不同的抽样,降低了算法对噪声的敏感度,提高了聚类质量.但这两类算法对数据的输入顺序较敏感,且只能处理凸状或球状的数据集.1996 年,Zhang 等人提出了 BIRCH 算法[14].该算法通过引入聚类特征(CF)和聚类特征树(CF 树)两个特有的概念,将原始数据集用树形结构压缩存储.新颖的数据结构不仅使算法节省了存储空间和运算量,而且也增强了对噪声的鲁棒性.但 BIRCH 算法同样对数据的输入顺序十分敏感,且对非凸状数据集不能很好地聚类.为使聚类算法能够处理任意形状的大型数据集,提出了基于多个代表点的 CURE 算法[15].在该算法中,选取一组固定数目且散布较好的点作为簇代表点,这使算法能够较为准确地捕获到簇的形状和大小.同时,通过代表点向簇质心的收缩,抑制了噪声点对聚类结果的影响,且增强了算法对不同簇稠密程度的辨识能力.面对任意形状的大型数

据集,CURE 算法首先对原始数据集进行随机抽样,然后对样本集进行划分,消除异常点,选取划分类别中局部的簇进行聚类,进而实现对整个数据集的最终聚类.CURE 算法虽然能够发现非凸集、差异较大的簇和异常点,但其收缩因子往往较难确定,过大、过小都会削弱算法的聚类优势.

通过对上述聚类算法的分析可知,选取样本集的质量以及对样本集的聚类情况决定了基于抽样的聚类算法对大规模数据聚类的有效性.在寻求更优的聚类算法的道路上,一种简单、快速的增量聚类算法——Leaders 算法[16−18]逐渐成为研究的热点.该算法通过选取每一簇的代表点 leader 来完成聚类,过程中虽忽略了数据点类别归属的模糊性,但其优势在于只需扫描一遍数据集,不需指定聚类个数,且只对选出的代表点 leader 进行存储.

大多数研究人员将 Leaders 算法作为预处理算法,并结合其他聚类算法用以处理大规模数据.例如,2004 年,Vijaya 等人在 Leaders 算法的基础上提出了层次聚类算法 Leaders-Subleaders 算法[16],在获得第 1 层 leader 后,继续利用 Leaders 算法对每一簇获得 subleader,有效地提高了算法在处理大型数据集时的聚类精度;针对 SVM 及 DBSCAN 算法不能很好地聚类大型数据集的问题,Enrique Romero 利用 Leaders 算法对数据集进行预处理,并在获得的具有代表性的训练集上应用 SVM 算法,在保证高分类准确率的情况下,明显地降低了算法的复杂度和运行时间[17];而 Viswanath 等人将 Leaders 算法与 DBSCAN 算法相结合,提出了针对大数据的 Rough-DBSCAN 算法[18].在该算法中,首先用固定 leader 产生顺序的 Leaders 算法对原始数据集进行初始划分,获得代表点集(leader 集)、每一类别包含的数据点及其个数;然后针对代表点集,应用 DBSCAN 算法进行聚类;在判断当前 leader 是否为稠密点时,单纯依靠其固定半径的邻域内 leader 的个数并不合理,因此,基于模糊集理论的 Rough-DBSCAN 算法将当前 leader 固定半径的邻域内所有 leader 代表类别内包含数据点的个数作为判断其稠密程度的依据.图 1 给出了判断稠密点时 leader 可能出现的 3 种位置情况.记 ld 为当前 leader,其ε邻域内的数据点由 3 部分组成:ld_1 及其代表类别内的所有数据点、ld_2 及其代表类别内的部分数据点、ld_3 代表类别内的部分数据点.Rough-DBSCAN 算法假设数据点在小范围内的分布是均匀的,用 ld_2 类别内阴影区域的数据点代替 ld_3 类别内阴影区域的数据点,即,只将满足$||ld_i–ld||\leqslant\varepsilon$的 leader 代表类别内的数据点计入$\varepsilon$邻域,近似合理地估计了当前 leader 的稠密程度.在代表点集聚类结束后,算法根据数据点与 leader 的对应关系完成对原始数据集的聚类.Rough-DBSCAN 算法在聚类效果上与 DBSCAN 算法十分接近,它不仅保留了 DBSCAN 算法的聚类优势,而且将算法的时间复杂度降低为 $O(n)$,非常有利于实际应用.

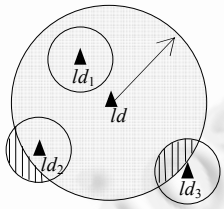


Fig.1 Three possible locations of leader in the neighborhood

图 1 邻域内 leader 可能出现的 3 种位置情况

受 Rough-DBSCAN 算法的启发,本文基于抽样思想提出了针对大数据的密度统计合并算法(DSML).该算法的主要特点是:将数据点的每个特征看作一组独立随机变量,并根据独立有限差分不等式获得统计合并判定准则.在聚类之前,DSML 算法利用统计合并判定准则对 Leaders 算法做出改进,并将改进后获得的 Statistical Leaders 算法作为自身的抽样算法.其聚类步骤如下:

1) 利用改进后的 Leaders 算法(statistical leaders)对原始数据集进行抽样,获得代表点集、每一 leader 代表类别内包含的数据点及其个数.
2) 确定每一 leader 在代表点集中的 k 邻域和密度信息,按照密度从大到小的顺序依次将 leader 同其 k 邻域内的点进行统计合并判定,实现对代表点集的聚类.
3) 依照 leader 的类别归属,将其代表类别内包含的数据点进行相同的归类,完成对整个数据集的聚类.

相对于前面介绍的 Rough-DBSCAN 算法,DSML 算法在聚类过程中不仅利用了数据点的密度信息,而且利

用了统计合并判定准则得出的数据点每一特征的差异性信息.因此,该算法对噪声具有更好的鲁棒性,对非凸形状或密度不均的大型数据集具有更好的聚类效果.同时,该算法还延续了抽样算法的聚类优势,运行时间要快于Rough-DBSCAN 算法,对处理大规模数据十分有利.

2 Statistical Leaders 算法

Statistical Leaders 算法作为 DSML 算法的抽样算法,是在改进 Leaders 算法的基础上得来的.

2.1 Leaders算法

Leaders 算法是一种增量聚类算法,其聚类结果取决于对代表点 leader 的选取.具体而言,该算法首先从数据集中任取一点作为起始 leader,然后依次计算剩余数据点与已有 leader 之间的欧式距离:若小于或等于给定阈值,则将数据点同当前 leader 归为一类;若大于给定阈值,则将数据点作为一个新的 leader.重复上述步骤,直到完成对整个数据集的聚类.过程细节见算法 1.

算法 1. Leaders.

输入:数据集 X;距离阈值 T.

1. 将 leader 的集合 L 初始化为空集,leader 的个数 l 初始化为 0;
2. **for** 数据集 X 中的每一个数据点 x **do**
3. 寻找一个 leader 满足:leader$\in L$ 且$||leader-x||\leqslant T$;
4. **if** (没有满足条件的 leader 或 $L=\varnothing$) **then**
5. $L=L\cup\{x\}$; $l=l+1$;
6. **else**
7. 将该数据点 x 归类到 leader 代表的类别中;
8. **end if**
9. **end for**

输出: leader 的集合 L;leader 的个数 l;聚类结果.

Leaders 算法只需扫描一遍数据集即可获得聚类结果,且在聚类过程中仅对选取的 leader 进行存储,大大节省了算法的运行时间和存储空间.时间复杂度和空间复杂度分别为 $O(n)$和 $O(l)$,对处理大规模数据和在线数据十分有利.此外,Leaders 算法也不需要指定聚类个数,通过调整参数 T 的大小,即可改变对数据集的分类细度.

作为一种简单、快速的聚类算法,除上述优势外,其自身也存在一些不足之处:首先,Leaders 算法的聚类结果受数据点输入顺序的影响,针对不同的输入顺序,算法会选取不同的 leader,从而产生不同的聚类结果;其次,在聚类结果中可能出现不同类别内两点相似性大于相同类别内两点相似性的情况,当输入顺序一致时,对于数据点 x,满足条件$||leader-x||\leqslant T$的 leader 可能不止一个,然而算法采用随机原则对满足条件的 leader 归类,导致聚类结果虽能保证同一类别内两个点之间的距离最大不超过 $2T$,却不能保证不同类别内两个点之间的距离大于 T.图 2 给出了两类别可能出现的不好的聚类情况.

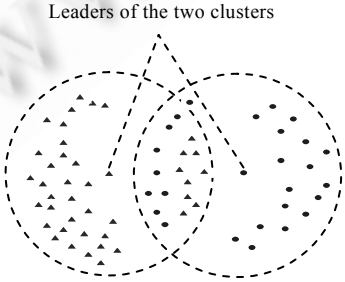


Fig.2 Bad clustering situation of Leaders algorithm

图 2 Leaders 算法中不好的聚类情况

2.2 Statistical Leaders算法

通过对 Leaders 算法的分析可知:图 2 中出现的聚类情况使每一类别内包含数据点的个数出现误差,不能正确反映 leader 在原始数据集中的密度信息(这里与 Rough-DBSCAN 算法相同,将每一 leader 的 k 邻域中所有 leader 代表类别内包含数据点的总个数作为其密度信息);又因 Leaders 算法在聚类过程中采用固定的距离阈值进行分类,使得选取的 leader 在原始数据集中分布较为均匀,因此,Leaders 算法作为抽样算法不能得到令人满意的抽样结果.

为使抽样获得的样本集更具代表性,并且能够较为准确地反映每一 leader 在原始数据集中的密度信息,在 Leaders 算法的基础上提出了一种结合统计思想的 Statistical Leaders 算法.该算法在进行聚类之前需要对数据集建立统计模型,并利用独立有限差分不等式得到特定的统计合并判定准则[19].

2.2.1 统计模型

Statistical Leaders 算法将数据点的每一个特征均用 Q 个独立随机变量表示(Q 的取值范围为正整数),且每一个随机变量对应一个分布.用 $\Omega=\{A,B,C,\ldots\}$ 表示特征集合,假设每个特征的取值范围为 $[L_i,U_i](i=A,B,C,\ldots)$,为方便应用,对数据集作整体移动,使得特征的取值范围变为 $[0,g_i](i=A,B,C,\ldots)$,其中,$g_i=|U_i-L_i|$,那么 Q 个独立随机变量和的取值也应属于 $[0,g_i](i=A,B,C,\ldots)$,每个随机变量的取值范围即为 $[0,g_i/Q](i=A,B,C,\ldots)$.这样,一个数据点的特征信息就由多组随机变量表示,数据集的统计模型由此建立.

该统计模型对数据点及数据点特征的取样是相互独立的,对 Q 个独立随机变量的分布也没有特定要求,即,独立不一定同分布.随机变量个数 Q 的传统取值一般为 1,但这一取值对相似性较高的数据集难以获得可靠的特征信息.当增大 Q 时,数据点的特征可被描述得更加细致,数据点之间的差异更加明显,使得数据集聚类时的类别个数会随之增加.因此,Q 成为调整 Statistical Leaders 算法聚类结果的重要参数.

基于该统计模型,理想的聚类结果应使得同一类别中的数据点对于任意数据特征均具有相同的期望,不同类别中的数据点至少有一个数据特征的期望不同.

2.2.2 统计合并判定准则

在 Statistical Leaders 算法中,数据点的合并由一个特定的统计合并判定准则决定.为简化准则的推导过程,先只考虑含有一个特征信息的数据集,即,一个数据点用一组独立随机变量表示.在此基础上,将得到的结果扩展到具有更多特征信息的数据集中.

首先介绍如下定理:

定理 1(独立有限差分不等式[19]). 设 $\boldsymbol{X}=(X_1,X_2,\ldots,X_n)$ 是一组独立随机变量,X_k 的取值范围为 $A_k(k=1,2,\ldots,n)$.设存在一个定义在 $\prod_k A_k$ 的实值函数 f,当变量 $\boldsymbol{X}$ 与 $\boldsymbol{X}'$ 仅在第 k 个条件不同时满足 $|f(\boldsymbol{X})-f(\boldsymbol{X}')|\leqslant r_k$,则 $\forall\tau\geqslant 0$,有:

$$P(f(X)-\mu\geqslant\tau)\leqslant\exp\left(-2\tau^2/\sum\nolimits_k (r_k)^2\right),$$

其中,μ 为 $f(\boldsymbol{X})$ 的期望,即,$\mu=Ef(\boldsymbol{X})$.

根据定理 1,结合建立的统计模型,可以推出数据集中不同类别绝对偏差不等式成立的概率范围.记 C 为数据集中的类别(单个数据点可作为一个类别),$|C|$ 为类别内数据点的个数,$\hat{C}$ 表示类别 C 与其他类别合并时的代表点,$E(C)$ 表示该类别相关数据点 Q 个独立随机变量期望和的期望.

推论 1. 考虑数据集中的类别组合 (C_1,C_2),$\forall 0<\delta\leqslant 1$,下面不等式成立的概率不超过 δ:

$$|(\hat{C}_1-\hat{C}_2)-E(\hat{C}_1-\hat{C}_2)|\geqslant g\sqrt{\frac{1}{2Q}\left(\frac{1}{|C_1|}+\frac{1}{|C_2|}\right)\ln\frac{2}{\delta}},$$

其中,$g=\max(g_i)(i=A,B,C,\ldots)$.

证明:已知类别 C_1 中的数据点可由 $Q|C_1|$ 个独立随机变量表示,类别 C_2 中的数据点可由 $Q|C_2|$ 个独立随机变量表示.$(\hat{C}_1-\hat{C}_2)$ 为实值函数,由于 $\hat{C}_1,\hat{C}_2$ 分别是 C_1,C_2 的代表点,若变动 C_1 中的变量,则 r_k 的最大取值为 $g/(Q|C_1|)$;若变动 C_2 中的变量,则 r_k 的最大取值为 $g/(Q|C_2|)$.记 $r_{C_1}=g/(Q|C_1|)$,$r_{C_2}=g/(Q|C_2|)$,则

$$\sum_k (r_k)^2 = Q(|C_1|(r_{C_1})^2 + |C_2|(r_{C_2})^2) = \frac{g^2}{Q}\left(\frac{1}{|C_1|}+\frac{1}{|C_2|}\right).$$

根据定理 1,取 $\tau = g\sqrt{\frac{1}{2Q}\left(\frac{1}{|C_1|}+\frac{1}{|C_2|}\right)\ln\frac{2}{\delta}} > 0$,则

$$P(|(\hat{C}_1-\hat{C}_2)-E(\hat{C}_1-\hat{C}_2)|\geqslant\tau)\leqslant\exp(-2\tau^2/\sum_k(r_k)^2)=\frac{\delta}{2}<\delta.$$

推论得证. □

由推论 1 可知:当δ取值接近于 0 时(本文若未特别标明,δ取为 $1/(6|X|^2)$),类别组合(C_1,C_2)满足不等式 $|(\hat{C}_1-\hat{C}_2)-E(\hat{C}_1-\hat{C}_2)|\leqslant b(C_1,C_2)$ 的概率接近于 1,其中,$b(C_1,C_2)=g\sqrt{\frac{1}{2Q}\left(\frac{1}{|C_1|}+\frac{1}{|C_2|}\right)\ln\frac{2}{\delta}}$;当 C_1 和 C_2 在理想的聚类结果中属于同一类别时,有 $E(\hat{C}_1-\hat{C}_2)=0$. 根据这两个条件可以得出:若类别组合(C_1,C_2)满足不等式 $|(\hat{C}_1-\hat{C}_2)|\leqslant b(C_1,C_2)$,则可以合并.因此,将 Statistical Leaders 算法的统计合并判定准则定义为

$$\mathcal{M}(C_1,C_2)=\begin{cases}\text{true}, & \text{if } |(\hat{C}_1-\hat{C}_2)|\leqslant b(C_1,C_2)\\ \text{false}, & \text{otherwise}\end{cases}.$$

将该准则扩展到具有多个特征信息的数据集中,形式如下:

$$\mathcal{M}(C_1,C_2)=\begin{cases}\text{true}, & \text{if } \forall a\in\{A,B,...\}, |(\hat{C}_{a1}-\hat{C}_{a2})|\leqslant b(C_1,C_2)\\ \text{false}, & \text{otherwise}\end{cases}.$$

即,当类别组合(C_1,C_2)对每个特征均满足不等式 $|(\hat{C}_{a1}-\hat{C}_{a2})|\leqslant b(C_1,C_2)$ 时,合并 C_1 和 C_2;反之则不然.

2.2.3 Statistical Leaders 算法的实现

获得统计合并判定准则后,Statistical Leaders 算法在保证 Leaders 算法聚类框架不变的情况下,将距离阈值用统计合并判定准则代替,同时借鉴 Rough-DBSCAN 算法中的改进,将 leader 产生的顺序固定为聚类顺序,以此获得更具代表性和密度信息更为准确的样本集.具体而言,该算法首先从数据集中任取一个数据点作为起始 leader,然后,依次将剩余数据点按聚类顺序与已有的 leader 进行统计合并判定:若满足判定准则,则将数据点合并到该 leader 代表的类别中,更新该类别包含的数据点 followers 及其个数 count;若不满足,则将数据点作为新的 leader,更新 leader 的个数.重复上述过程,直到对整个数据集完成初始分类,获得代表点集.过程细节见算法 2.

算法 2. Statistical Leaders.

输入:数据集 X;独立随机变量个数 Q.

1. 将 leader 的集合 L 初始化为空集,leader 的个数 l 初始化为 0;
2. **for** 数据集 X 中的每一个数据点 x **do**
3. 寻找一个 leader 满足:leader$\in L$,代表类别 C_{ld} 且$\forall a\in\{A,B,\ldots\}$,$|(\text{leader}_a-x_a)|\leqslant b(C_{ld},x)$;
4. **if** (没有满足条件的 leader 或 $L=\varnothing$) **then**
5. $L=L\cup\{x\}$; $l=l+1$;
6. $followers(x)=\{x\}$; $count(x)=1$;
7. **else**
8. 若有多个满足条件的 leader,按聚类顺序选择第一个满足条件的 leader;
9. $followers(leader)=followers(leader)\cup\{x\}$; $count(leader)=count(leader)+1$;
10. **end if**
11. **end for**

输出:leader 的集合 L;leader 的个数 l;每个 leader 的 followers 和 count.

与 Leaders 算法选取固定的距离阈值不同,Statistical Leaders 算法选取的统计合并判定准则的右边阈值 $b(C_1,C_2)$与类别中包含数据点的个数相关,会随着类别内数据点个数的增加而减小,因此使得选取 leader 的数目

在稠密区域增长较快,在稀疏区域增长较慢,更能反映原始数据集的稠密程度,提高了样本集的代表性.同时,Statistical Leaders 算法按固定的聚类顺序选择满足条件的 leader 合并,有效地避免了图 2 出现的聚类问题,使获得的代表点的密度信息更接近于其真实密度.进一步来说,由于统计合并判定准则的计算在速度上要快于距离计算,因此与 Leaders 算法相比,Statistical Leaders 算法在运行时间上更具优势.

2.3 Statistical Leaders算法的实验分析

在本节中,对 Statistical Leaders 算法进行了仿真实验.实验目的为:1) 分析 Statistical Leaders 算法参数的性质;2) 证明 Statistical Leaders 算法抽样的有效性.实验对象选取二维空间上 3 个不同类型的人工数据集.表 2 给出了数据集的基本信息,图 3 显示了对应数据集的几何形状,它们均是通过不同参数的高斯分布随机生成.Data Set 1 由 3 个 circle 组成,每个 circle 包含相同数目的数据点,属于凸状数据集;Data Set 2 由一个 ring 和两个 circle 组成,其中两个 circle 的数据量之比固定为 1:5,整体含有 1 000 个随机噪声点,是一种凸状与非凸状数据集的混合;Data Set 3 由 10 个大小环组成,外环半径为 3,两个外环圆心的水平距离为 9,同时,外环与内环的数据量之比固定为 3:1,整体含有 1 000 个随机噪声点,属于非凸状的数据集.实验的所有程序采用 MATLAB 编写,并在配置为 Intel(R) Core(TM) i3~2130 CPU @ 3.40Hz,3.4GB 内存的计算机上运行.

Table 2 Basic information of artificial data sets

表 2 人工数据集的基本信息

数据集	数据量	含随机噪声点数	数据集形状	类别数
Data Set 1	120 000	0	Circles of equal sizes	3
Data Set 2	85 000	1 000	Rings and circles	3
Data Set 3	200 000	1 000	Big and small rings	10

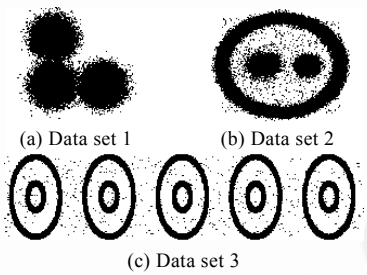


Fig.3 Artificial data sets

图 3 人工数据集

2.3.1 Statistical Leaders 算法的参数分析

Statistical Leaders 算法的参数为独立随机变量的个数 Q.它不仅可以控制算法的抽样精度,还可以改变算法的统计复杂度.

图 4 显示了数据集在固定数据量的情况下,随着参数 Q 的增大,抽样 leader 个数的变化情况.可以看出,对 3 个不同类型的数据集,参数 Q 的取值越大,抽取 leader 的数量越多.这是因为当每个特征用更多的独立随机变量表示时,特征差异会被描述的更加细致,导致数据集的类别个数增多.而从统计合并判定准则的角度来讲,Q 增大,导致准则右边的阈值 $b(C_1,C_2)$变小,因此,抽样个数会明显增多.将参数 Q 的取值从小调大,可以得到 leader 由少到多的代表点集.

从图 4 还可以看出,相同的 Q 值对 3 个不同的数据集抽取的 leader 个数不同.这与数据集的大小有关,还是与数据集的空间分布有关呢?将 3 个数据集的数据量在固定高斯分布的基础上等量增多,利用固定的参数 Q 对同类不同量的数据集进行抽样,观察抽样个数的变化情况.从图 5(K=10^3)的实验结果可以看出,对于 3 个不同的数据集,在抽取 leader 个数小于 4 000 时,固定的 Q 值对数据量逐渐增加的同类数据集抽取的 leader 个数十分接近;而对比 3 幅图中数据量相近的不同数据集,相同的 Q 值对它们的抽样各不相同.这就说明,当 Statistical

Leaders 算法用相同的 Q 值对不同的数据集进行抽样时,抽取 leader 的个数受数据集规模的影响较小,主要与数据集的空间分布密切相关,即,对分布较为紧凑的数据集(如 Data Set 1 和 Data Set 2)抽取 leader 较多,对分布较为离散的数据集(如 Data Set 3)抽取 leader 较少.

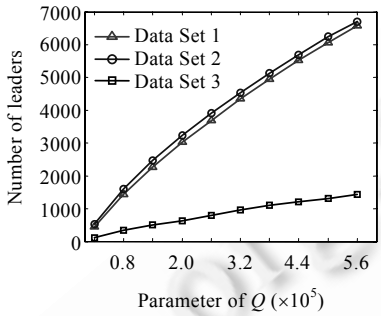
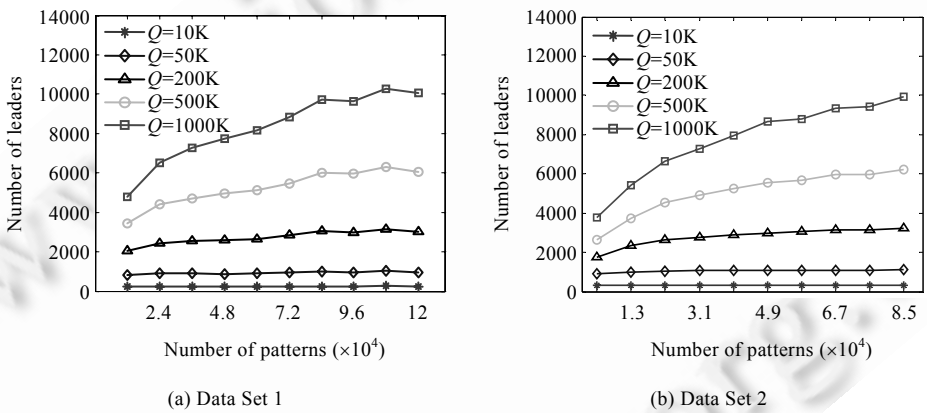
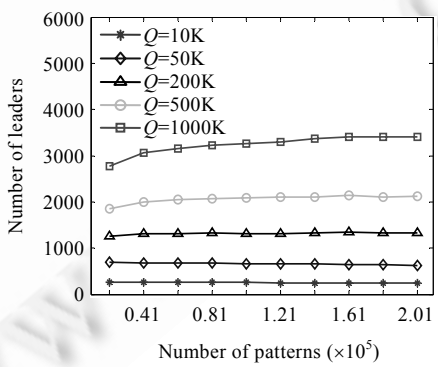


Fig.4 Influence of parameter Q on leader's number

图 4 参数 Q 对抽样 leader 个数的影响



(a) Data Set 1 (b) Data Set 2



(c) Data Set 3

Fig.5 Characteristic description of real data sets

图 5 针对同类不同量的数据集的抽样

了解了参数 Q 的基本性质,下面分析 Q 的取值对 Statistical Leaders 算法抽样性能的具体影响.

图 6 以 Data Set 1 为例,给出了增大 Q 时,Statistical Leaders 算法抽取 leader 增多的具体情况.可以看出,当 Q

的取值较小时,抽取 leader 的个数较少且分布均匀;随着 Q 的增大,抽取 leader 个数增多,代表点集对原始数据集稠密程度的反映变得更加准确.

(a) Q=20000, l=441 (b) Q=140000, l=2254 (c) Q=320000, l=4337

Fig.6 Concrete condition of leader with increased Q

图 6 随着 Q 的增大,leader 增多的具体情况

图 7 给出了参数 Q 增大时,Statistical Leaders 算法所需时间的具体情况.可以看出,随着 Q 的增大,抽取 leader 的个数逐渐增多,Statistical Leaders 算法所需的运行时间越来越长.

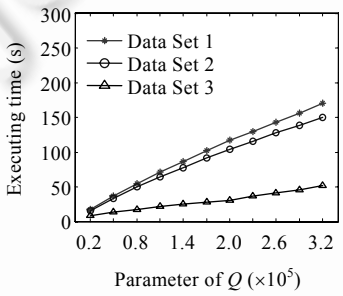


Fig.7 Changes of running time with increased Q

图 7 随着 Q 的增大,算法运行时间的变化情况

综合图 6 和图 7 的实验结果可知:当 Q 的取值较小时,Statistical Leaders 算法所需的运行时间较少,但可能抽取的 leader 分布过于均匀,不足以反映原始数据集的稠密情况;当 Q 的取值较大时,Statistical Leaders 算法抽取的 leader 能够较好地反映原始数据集的密度信息,但算法运行花费的时间过长.因此,在选取参数 Q 时,要同时兼顾算法的抽样效果及运行时间.Statistical Leaders 算法一般默认 Q 的初始值为 10 000,在具体应用中,根据数据集的空间分布和数据点的差异程度进行调整.

2.3.2 Statistical Leaders 算法与 Leaders 算法的对比分析

为了证明 Statistical Leaders 算法抽样的有效性,本节将 Statistical Leaders 算法与 Leaders 算法做对比分析,对比的重点为算法抽取样本集的代表性和算法的运行时间.这里将算法抽取的非噪声点个数占代表点总个数的比值作为样本集代表性的评价指标,记作 P.P 值越大,说明算法抽取样本集的代表性越高,抽样算法就越好.实验对象是在保持 Data Set 2 高斯分布不变的情况上,将数据量降低到 30 000 获得的数据集,含 1 000 个随机噪声点.在实验过程中,对固定的数据集分别应用 Leaders 算法和 Statistical Leaders 算法进行多次抽样,并保证每次抽样时两者得到的 leader 个数尽可能地接近,记录抽取 leader 时两者对应的 P 值和时间,具体见表 3.

从表 3 中可以看出,当两种算法抽取的 leader 个数接近时,Statistical Leaders 算法的 P 值总是高于 Leaders 算法的 P 值,且所需的运行时间只有 Leaders 算法的 1/2 左右.说明两种算法在抽取相近大小的代表点集时,Statistical Leaders 算法的代表点集对原始数据集的稠密信息反映更好,抽样的执行速度更快.这充分表明了 Statistical Leaders 算法抽样的有效性.

Table 3 Comparison of the sampling performance between Leaders and Statistical Leaders algorithm

表 3 Leaders 算法与 Statistical Leaders 算法抽样性能的对比

Leaders				Statistical Leaders			
T	l	P	t (s)	$Q\times10^3$	l	P	t (s)
0.17	395	0.650 6	10.75	11	384	0.656 3	4.11
0.095	1 142	0.602 5	23.48	40	1 139	0.611 1	11.10
0.065	2 128	0.591 2	43.40	90	2 095	0.615 3	20.70
005	3 210	0.589 1	65.62	165	3 199	0.616 8	32.64
0.042	4 150	0.595 7	87.07	240	4 108	0.620 0	41.63
0.035	5 264	0.607 9	116.24	360	5 246	0.629 2	55.37
0.031 5	6 009	0.616 6	139.19	450	6 007	0.633 6	64.70
0.027 5	7 049	0.627 6	164.28	620	7 093	0.646 3	75.79
0.024 5	8 018	0.640 8	194.14	800	8 097	0.654 4	87.72

3 DSML 算法

针对大规模数据的聚类问题,将 Statistical Leaders 算法作为抽样算法,并结合 leader 的密度信息和统计合并判定准则,提出了大数据的密度统计合并算法——DSML 算法.该算法的显著优势为:(1) 可以聚类任意形状的大型数据集;(2) 对噪声具有良好的鲁棒性;(3) 算法的时间复杂度为 $O(n)$,执行效率较高.

3.1 DSML算法的聚类过程

DSML 算法的聚类框架如图 8 所示.

- 第 1 阶段:抽样

利用 Statistical Leaders 算法对原始数据集进行抽样,获得代表点集、每一 leader 代表类别内包含的数据点及其个数.抽样过程建立了一个从原始数据集到代表点集的映射,每个数据点在代表点集中都有唯一一个 leader 与之相对应,数据点最终的类别归属由对应 leader 直接决定.

- 第 2 阶段:对代表点集聚类

首先,通过计算 leader 之间的欧式距离构建相似性矩阵,选取与每一 leader 相似性最高的前 k 个点组成 k 邻域;然后,将 leader 的密度信息从大到小排序,以此作为聚类时的合并顺序.这里将每一 leader 的 k 邻域中所有 leader 代表类别内包含数据点的总个数作为其密度信息,数据点个数越多,密度越大.按照该合并顺序依次将 leader 与其 k 邻域内的点进行统计合并判定,若满足判定准则,则将该点与 leader 合并.遍历所有的 leader,最终完成对代表点集的聚类.在这一阶段,基于密度的合并顺序保证了在任意两个不同的类别进行合并判定时,其自身已经完成了所有可能的合并.

- 第 3 阶段:根据映射关系完成对整个数据集的聚类

由于代表点集与原始数据集之间建立了映射,在完成对代表点集的分类后,依据 leader 的类别归属,将与其具有对应关系的数据点进行相同的归类,即实现了对整个数据集的聚类.

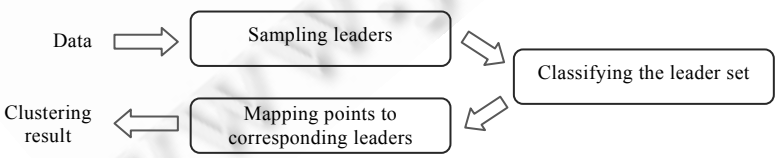


Fig.8 Frame of the DSML algorithm

图 8 DSML 算法框架

DSML 算法在聚类过程中分别对原始数据集和代表点集建立统计模型,使用两次统计合并判定准则.第 1 次改进抽样算法获得代表点集,第 2 次结合 leader 的密度信息对其 k 邻域进行聚类,因此,DSML 算法的聚类效果由 3 个参数决定:抽样时独立随机变量的个数 Q_1、针对代表点集聚类时独立随机变量的个数 Q_2、邻域内数

据点的个数 k.由于调整 Q 的大小可以改变算法的分类细度,即,当 Q 较小时,数据集分类较少;当 Q 较大时,数据集分类较多,所以作为抽样算法的参数 Q_1,其取值应远大于参数 Q_2.

3.2 DSML算法的实现

为更好地处理噪声点,DSML 算法在对代表点集进行聚类时,只对合并顺序前α比例的 leader 进行统计合并判定(本文默认α=0.9),剩余的 leader 根据临近原则进行归类.下面具体说明 DSML 算法的实现细节.

算法 3. DSML.

输入:数据集 X;独立随机变量个数 Q_1 和 Q_2;邻域内数据点个数 k.

1. 调用 Statistical Leaders 算法,获得代表点集 L,leader 个数 l,*followers*(*leader*),*count*(*leader*);
2. 对每一 leader 赋予互不相同的类标号 N,对应表示不同类别 C_N;
3. 计算代表点集 L 中任意两个 leader 之间的欧氏距离,确定每一 leader 的 k 邻域;
4. 计算每一 leader 的 k 邻域中包含 leader 的 *count*(*leader*)总数,按降序排列,获得 leader 的合并顺序 d;
5. **for** 代表点集 L 中 90%的 leader **do**
6. 按合并顺序 d 找到当前 leader,确定类标号为 N_l;
7. **for** 当前 leader 的 k 邻域中每一点 y **do**
8. **if** ($N_l \neq N_y$ 且满足$\forall a \in \{A,B,\ldots\}, |(leader_a - y_a)| \leqslant b(C_{N_l} - C_{N_y})$) **then**
9. 将 C_{N_l} 与 C_{N_y} 合并,N_l 与 N_y 两个类别标号统一;
10. **end if**
11. **end for**
12. **end for**
13. **for** 代表点集 L 中剩余的 10%的 leader **do**
14. 按合并顺序 d 找到当前 leader,确定类标号为 N_l;
15. 确定当前 leader 的 k 邻域中的每一点 y 的类标号;
16. **if** (存在类标号 N_y 满足: C_{N_y} 至少包含两个点且 leader 的 k 邻域中被标记 N_y 的点数最多) **then**
17. $C_{N_y} = C_{N_y} \cup \{leader\}$, $N_l = N_y$;
18. **end if**
19. **end for**
20. 根据 leader 最终的类标号将 *followers*(*leader*)归入相应类别;
21. 计算不同类标号的个数 nbcluster;

输出:聚类结果;聚类个数 nbcluster.

3.3 时空复杂度分析

假设数据集中包含数据点个数为 n,抽样选取 leader 个数为 l,邻域内数据点个数为 k.由 DSML 算法的聚类过程可知,其计算量主要集中于以下 3 个步骤:

1) Statistical Leaders 算法对原始数据集的抽样;
2) 构建 leader 的距离度量矩阵,获得每一 leader 的 k 邻域;
3) 统计合并判定时,对 leader 及其 k 邻域内的点进行迭代.

对于步骤 1),Statistical Leaders 算法与 Leaders 算法相同,只对数据集扫描一遍,且仅对选取的 leader 进行存储,因此,对给定的含有 n 个点的数据集,选取 l 个 leader 的时间复杂度为 $O(n)$,空间复杂度为 $O(l)$;对于步骤 2),针对含有 l 个 leader 的代表点集,计算距离度量矩阵的时间复杂度为 $O(l^2)$,之后对获得的 leader 的 k 邻域进行存储,空间复杂度为 $O(lk)$;对于步骤 3),将当前 leader 与其 k 邻域中的点进行统计合并判定时,k 邻域内点的最大迭代次数为 k,因此,步骤 3)遍历完代表点集中所有 leader 的时间复杂度不超过 $O(lk)$.综合上面的分析可知,DSML 算法的时间复杂度为 $O(n+l^2+lk)$,空间复杂度为 $O(l)+O(lk)$.但因 leader 的个数 l 有独立于数据集个数和分布的上

限[19],且处理大数据集时 l 和 k 的取值通常远小于 n,因此,DSML 算法的时间复杂度可近似于抽样算法的时间复杂度 $O(n)$,具有较高的执行效率.

3.4 DSML算法的实验分析与评价

在本节中,为了更加直观地说明 DSML 算法对大规模数据聚类的有效性,选取具有相似聚类框架的 Rough-DBSCAN 算法进行对比分析.实验环境与第 2.3 节相同,实验对象选取第 2.3 节中的 3 个人工数据集及部分真实数据集.第 3.4.1 节是对人工数据集的聚类结果对比;第 3.4.2 节是对真实数据集的聚类结果对比;第 3.4.3 节是对 DSML 算法中参数的讨论分析;第 3.4.4 节是对两种算法执行效率的对比.

实验过程中,由于聚类结果的好坏取决于算法对数据集中非噪声点的分类情况,因此,将算法聚类的评价指标定义为非噪声点的分类准确率,记为 F.另外,由于 Rough-DBSCAN 算法与 DSML 算法具有互不相同的参数,对实验中参数的选取做以下说明:

- Rough-DBSCAN 算法:共有 3 个参数,分别是抽样时距离阈值 T、针对代表点集聚类时的邻域半径 r 及邻域内点的个数阈值 m.对大规模数据,参数 m 一般取为 100 左右,参数 T 和 r 则根据数据集的直径做决定.
- DSML 算法:共有 3 个参数,分别是抽样时独立随机变量的个数 Q_1、针对代表点集聚类时独立随机变量的个数 Q_2、邻域内数据点的个数 k.在实验中,参数 k 一般取为 10 左右的数.对于该算法特有的参数 Q_1 和 Q_2,它们控制了算法的抽样精度和聚类细度.对大规模数据,参数 Q_1 的默认值为 10 000,具体根据数据集的空间分布和数据点的差异程度进行调整.参数 Q_2 要远小于 Q_1,取值范围为 100~5 000,具体根据数据集的分类需求进行调整.

3.4.1 人工数据集的聚类效果对比

对任意形状的数据集都有良好聚类效果的算法才能称为较好的聚类算法.在本节中,将 Rough-DBSCAN 算法与 DSML 算法同时应用于 3 个不同类型的人工数据集,在基于较好的抽样效果的前提下,对两种算法的聚类结果进行对比分析.

从表 4 的实验结果可以看出,DSML 算法在处理凸状的 Data Set 1 和混合状含噪声点的 Data Set 2 时,聚类效果要优 Rough-DBSCAN 算法;同时,在处理非凸状的 Data Set 3 时,聚类效果与 Rough-DBSCAN 算法持平.这充分表明了 DSML 算法不仅可以处理任意形状的数据集,而且对混合状、含噪声的数据集也具有良好的聚类效果.

Table 4 Comparison of clustering results between Rough-DBSCAN and DSML algorithm

表 4 Rough-DBSCAN 算法与 DSML 算法的聚类效果对比

数据集	Rough-DBSCAN					DSML				
	T	r	m	l	F	$Q_1\times10^4$	Q_2	k	l	F
Data Set 1	0.25	650	0.8	1 163	0.994 2	6	500	10	1 121	0.998 0
Data Set 2	0.09	300	0.25	1 106	0.941 0	5	200	10	1 121	0.988 1
Data Set 3	0.3	100	0.4	1 149	0.995 0	40	5 000	10	1 137	0.995 0

3.4.2 真实数据集的聚类效果对比

基于对人工数据集良好的聚类效果,本节将继续应用 DSML 算法对部分真实数据集进行聚类实验,并将聚类结果与 Rough-DBSCAN 算法做比较.实验对象选自 USPS 数据库(the United States Postal Service,由 16×16 维灰度图像构成,每个样本表示为 256 维向量,共包含 7 291 个训练样本、2 007 个测试样本)中的训练样本集和 UCI 数据库(http://archive.ics.uci.edu/ml/,加州大学欧文分校提出的用于机器学习的数据库,目前包含 223 个数据集)中的 letter 手写字母识别数据集.具体选取真实数据集的基本特征见表 5.

通过表 6 可以看出,DSML 算法对 7 种真实数据集的聚类效果都要优于 Rough-DBSCAN 算法,说明基于密度的 DSML 算法对真实数据集也具有良好的聚类效果.这进一步证明了 DSML 算法在聚类上的有效性.

Table 5 Characteristic description of real data set

表 5 真实数据集的特征描述

数据集	样本点数	特征个数	类别数
usps(1,7)	1 650	256	2
usps(0,6)	1 858	256	2
usps(0,1,5)	2 755	256	3
usps(0,1,2,3)	3 588	256	4
letter(A,H)	1 523	16	2
letter(P,R)	1 561	16	2
letter(A,B,C)	2 291	16	3

Table 6 Comparison of clustering results between Rough-DBSCAN and DSML algorithm

表 6 Rough-DBSCAN 算法与 DSML 算法的聚类效果对比

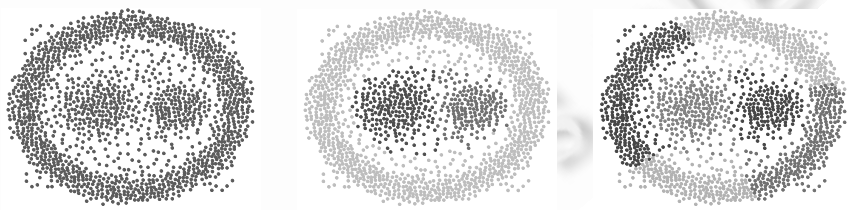
数据集	Rough-DBSCAN					DSML				
	T	r	m	l	F	Q_1	Q_2	k	l	F
usps(1,7)	4	6.3	5	636	0.900 6	30	1	6	579	0.990 3
usps(0,6)	7.5	8.75	45	613	0.748 7	15	1	16	539	0.958 6
usps(0,1,5)	7.7	8.61	10	784	0.722 0	18	1	7	728	0.909 3
usps(0,1,2,3)	8.5	9.21	3	1 056	0.716 6	16	1	6	983	0.810 5
letter(A,H)	2.5	4	3	648	0.698 0	1 000	10	7	471	0.940 2
letter(P,R)	2.6	4.2	33	729	0.668 8	2 500	10	8	614	0.923 1
letter(A,B,C)	2.7	4.3	3	792	0.865 6	2 500	20	10	795	0.928 4

3.4.3 DSML 算法中参数的讨论分析

DSML 算法的聚类效果由 3 个参数决定:抽样时独立随机变量的个数 Q_1、针对代表点集聚类时独立随机变量的个数 Q_2、邻域内数据点的个数 k.

Q_1 作为抽样算法的参数,已在第 2.3.1 节进行了详细的讨论.在本节实验中,首先利用 Statistical Leaders 算法(Q_1=80000)对数据集进行抽样,然后,在抽样得到的代表点集上对参数 Q_2 和 k 的性质进行实验分析.

参数 Q_2 的取值控制了算法对数据集的最终聚类个数.图 9 以 Data Set 2 为例,给出了在固定 k=10 的情况下,随着 Q_2 的变化,DSML 算法对代表点集的聚类情况.可以看出,随着参数 Q_2 逐渐增大,聚类个数逐渐增多.当 Q_2 取 300 左右时,可获得满意的聚类效果.



(a) Q_2=1 (1~110) (b) Q_2=300 (237~303) (c) Q_2=1000 (304 以上)

Fig.9 Influence of Q_2 on clustering result when k=10

图 9 当 k=10 时,Q_2 的取值对聚类结果的影响

图 10 给出了在固定 k=10 的情况下,随着 Q_2 的变化,DSML 算法对代表点集聚类时所需时间的变化情况.可以看出,随着 Q_2 的增大,DSML 算法聚类所需的时间并没有很大的变化.因此,参数 Q_2 的取值对算法运行时间没有特别的影响.

参数 k 的取值决定了算法的合并顺序.由于改变 k 的取值时,合并顺序会发生变化,所以不能完全确定参数 k 与聚类结果的稳定关系.但一般来说,随着 k 的增大,k 邻域内可以合并点数增多,会使聚类时得到的类别个数变少.图 11 给出了固定 Q_2=200 时,随着 k 的变化,DSML 算法对代表点集聚类时所需时间的变化情况.可以看出,随着 k 的增大,聚类所需的时间呈上升趋势.但由于整体时间很少,因此对整个 DSML 算法的运行时间并没有直接的影响.

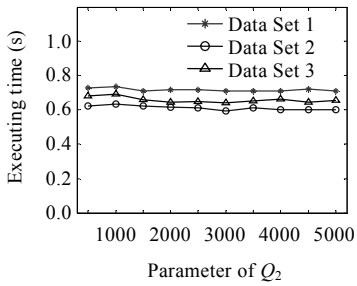


Fig.10 Relationship between Q_2 and time when k=10

图 10 k=10 时,Q_2 的取值与时间的关系

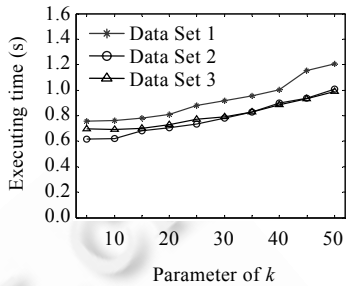


Fig.11 Relationship between k and time when Q_2=200

图 11 Q_2=200 时,k 的取值与时间的关系

综合上述对 DSML 算法参数的分析可知,DSML 算法的最终聚类结果由 3 个参数的共同作用决定,而 DSML 算法的运行时间主要由参数 Q_1 的取值决定.

3.4.4 执行效率的对比

在本节中,将 DSML 算法与 Rough-DBSCAN 算法的执行效率进行对比.实验对象选取两种算法对其具有相同聚类结果的 Data Set 3.实验过程中,为充分显示两种算法的最优执行效率,在保证算法得到最好聚类效果的同时,尽量减少 leader 的抽取个数.

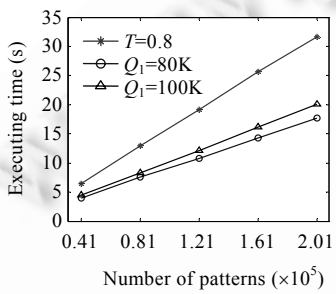


Fig.12 Comparison of execution efficiency

图 12 执行效率的对比

图 12 给出了数据规模增大时,两种算法执行效率的变化情况.其中,Rough-DBSCAN 算法的参数 T 取为 0.8 时,与 DSML 算法参数 Q_1 取为 80 000 时抽取 leader 的个数相近.比较算法此时的执行效率发现,DSML 算法在得到与 Rough-DBSCAN 算法相同聚类效果的情况下,所需运行时间只有 Rough-DBSCAN 算法的 1/2 左右.这充分证明了 DSML 算法在执行效率上的优势.进一步地,将 DSML 算法的参数 Q_1 增加到 100 000,再与 Rough-DBSCAN 算法进行对比发现,DSML 算法在抽取 leader 个数增多、抽样时间增长的情况下,整体运行时间依然比 Rough-DBSCAN 算法要快很多.这再次说明 DSML 算法是一种快速、高效的聚类算法.

4 总 结

随着信息技术的迅猛发展,大规模数据集层出不穷,这对聚类算法的执行效率有了更高要求.本文从提高算法的可扩展性和高效性的角度出发,结合抽样思想和统计思想,提出了一种针对大数据的密度统计合并算法——DSML 算法.该算法的突出特点是:将数据点的每一个特征看作一组独立随机变量,并根据独立有限差分不等式得到了特定的统计合并判定准则.在聚类过程中,DSML 算法首先利用统计合并判定准则对 Leaders 算法做出改进,并将改进后的算法作为抽样算法获得代表点集;然后,结合代表点的密度信息和邻域信息,再次利用统计合并准则完成对整个数据集的聚类.DSML 算法不仅保持了基于密度的聚类算法的优势,即,可以聚类任意形状的数据集及对噪声具有一定的鲁棒性,而且还有近似线性的时间复杂度,非常适合处理大规模数据集.与 Rough-DBSCAN 算法对比的实验结果充分说明了 DSML 算法的适用性和有效性.

References:

[1] Xu R, Wunsch D. Survey of clustering algorithms. IEEE Trans. on Neural Networks, 2005,16(3):645−678. [doi: 10.1109/TNN.2005.845141]

[2] Everitt BS, Landau S, Leese M, Stahl D. Cluster Analysis. 5th ed., London: John Wiley & Sons, Ltd., 2011.

[3] Jain AK, Murty MN, Flynn PJ. Data clustering: A review. ACM Computing Surveys, 1999,31(3):264−323. [doi: 10.1145/331499.331504]

[4] MacQueen JB. Some methods for classification and analysis of multivariate observations. In: Proc. of the 5th Berkeley Symp. on Mathematical Statistics and Probability. 1967. 281−297. [doi: 10.2307/1377831]

[5] Park HS, Jun CH. A simple and fast algorithm for *K*-medoids clustering. Expert Systems with Applications, 2009,36(2):3336−3341. [doi: 10.1016/j.eswa.2008.01.039]

[6] Kaufman L, Rousseeuw PJ. Finding Groups in Data: An Introduction to Cluster Analysis. New Jersey: John Wiley & Sons, 1990. [doi: 10.1002/9780470316801]

[7] Alpert CJ, Kahng AB. Recent directions in netlist partitioning: A survey. Integration, the VLSI Journal, 1995,19(1):1−81. [doi: 10.1016/0167-9260(95)00008-4]

[8] Ester M, Kriegel H, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proc. of the 2nd Int'l Conf. on Knowledge Discovery and Data Mining. Beijing: The AAAI Press, 1996. 226−231.

[9] Ankerst M, Breunig MM, Kriegel HP, Sander J. OPTICS: Ordering points to identify the clustering structure. ACM SIGMOD Record, 1999,28(2):49−60. [doi: 10.1145/304182.304187]

[10] Zhou S, Zhao Y, Guan J, Huang J. A neighborhood-based clustering algorithm. In: Proc. of the 9th Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD 2005). LNAI 3518, Berlin, Heidelberg: Springer-Verlag, 2005. 361−371. [doi: 10.1007/11430919_43]

[11] Rodriguez A, Laio A. Clustering by fast search and find of density peaks. Science, 2014,344(6191):1492−1496. [doi: 10.1126/science.1242072]

[12] Shirkhorshidi AS, Aghabozorgi S, Wah TY, Herawan T. Big data clustering: A review. In: Proc. of the Computational Science and Its Applications (ICCSA 2014). Springer-Verlag, 2014. 707−720. [doi: 10.1007/978-3-319-09156-3_49]

[13] Ng R, Han J. CLARANS: A method for clustering objects for spatial data mining. IEEE Trans. on Knowledge and Data Engineering, 2002,14(5):1003−1016. [doi: 10.1109/TKDE.2002.1033770]

[14] Zhang T, Ramakrishnan R, Livny M. BIRCH: An efficient data clustering method for very large databases. In: Proc. of the ACM SIGMOD Conf. on Management of Data. ACM Press, 1996. 103−114. [doi: 10.1145/233269.233324]

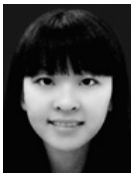
[15] Guha S, Rastogi R, Shim K. CURE: An efficient clustering algorithm for large databases. In: Proc. of the ACM SIGMOD. New York: ACM Press, 1998. 73−84. [doi: 10.1145/276304.276312]

[16] Vijaya PA, Narasimha MM, Subramanian DK. Leaders-Subleaders: An efficient hierarchical clustering algorithm for large data sets. Pattern Recognition Letters, 2004,25(4):505−513. [doi:10.1016/j.patrec.2003.12.013]

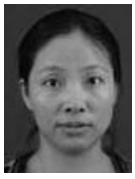
[17] Romero E. Using the leader algorithm with support vector machines for large data sets. Lecture Notes in Computer Science, 2011, 6791:225−232. [doi: 10.1007/978-3-642-21735-7_28]

[18] Viswanath P, Babu VS. Rough-DBSCAN: A fast hybrid density based clustering method for large data sets. Pattern Recognition Letters, 2009,30(16):1477−1488. [doi: 10.1016/j.patrec.2009.08.008]

[19] Nock R, Nielsen F. Statistical region merging. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2004,26(11):1452−1458. [doi: 10.1109/TPAMI.2004.110]



刘贝贝(1990－),女,山东新泰人,硕士生,主要研究领域为模式识别.



丁军娣(1978－),女,博士,副教授,CCF 会员,主要研究领域为模式识别,计算机视觉.

马儒宁(1976－),男,博士,副教授,主要研究领域为应用数学,模式识别,图像处理.