软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2013,24(3):490−506 [doi: 10.3724/SP.J.1001.2013.04322]

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

多层核心集凝聚算法*

马儒宁[1], 王秀丽[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: 丁军娣, E-mail: dingjundi2010@njust.edu.cn, http://www.patternrecognition.cn/~dingjundi

摘 要: 许多经典的聚类算法,如平均链接,*K*-means,*K*-medoids,Clara,Clarans 等,都是利用单一的聚类中心进行聚类.为克服单一聚类中心只能描述凸状聚类的缺陷,CURE,DBSCAN 等算法使用多个代表点(或稠密点)表述任意形状的聚类结构,但仍难以聚类重叠和噪声数据.为此,提出一种基于多层聚类中心(称为核心集)的凝聚聚类算法(MulCA).该算法使用了多层核心集表述聚类结构,使得每一层数据集向其核心集凝聚.同时,上层的核心集自动成为下层的数据集.随着每层核心集规模按α比例迅速减少,控制了凝聚过程的迭代次数.此外,引入了基于随机采样计算ε-核心集(RBC)的技巧,将 MulCA 算法应用于大规模数据集.大量的数值实验充分验证了 MulCA 算法的有效性.

关键词: 多层;核心集;凝聚;大规模

中图法分类号: TP181 **文献标识码**: A

中文引用格式: 马儒宁,王秀丽,丁军娣.多层核心集凝聚算法.软件学报,2013,24(3):490−506. http://www.jos.org.cn/1000-9825/4322.htm

英文引用格式: Ma RN, Wang XL, Ding JD. Multilevel core-sets based aggregation clustering algorithm. Ruanjian Xuebao/ Journal of Software, 2013,24(3):490−506 (in Chinese). http://www.jos.org.cn/1000-9825/4322.htm

Multilevel Core-Sets Based Aggregation Clustering Algorithm

MA Ru-Ning[1], WANG Xiu-Li[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)

Corresponding author: DING Jun-Di, E-mail: dingjundi2010@njust.edu.cn, http://www.patternrecognition.cn/~dingjundi

Abstract: Many classical clustering algorithms like Average-link, *K*-means, *K*-medoids, Clara, Clarans and so on are all based on a single cluster-center and are only apt to discover convex-structured clusters. Other methods, e.g., CURE and DBSCAN, use more than one point to represent a cluster and can find some well-separated clusters of arbitrary shape. However, they only consider the original scale of the input data; thus, they cannot depart over-lapped or noisy clusters. To this end, this paper is used to propose a multilevel core-set based agglomerative clustering algorithm (MulCA). The idea of MulCA is that the clustering structure is described by multi-level core set. Clustering process is achieved through procedure which the top of the core set automatically becomes the underlying data set. In addition, through the introduction of random sampling based ε-core set (RBC), MulCA algorithm is applied to large-scale data sets. A large number of numerical experiments fully verify the algorithm MulCA.

Key words: multilevel; core-set; aggregation; large size

聚类是一种重要的无监督学习过程,在模式识别、数据挖掘、机器学习、信息处理等领域具有重要应用.直观地讲,聚类就是在多维空间中将某给定的数据集划分为若干互不相交的子集,该过程不依赖于数据关于类别属性的先验信息,只考虑数据本身的特征属性以及数据之间的相互关系(度量或相似性)[1,2].整个聚类过程包

* 基金项目: 国家自然科学基金(61103058, 61233011, 61272220)
收稿时间: 2012-03-07; 修改时间: 2012-06-29; 定稿时间: 2012-09-12

括 3 个层次[3]:(a) 特征抽取,即得到原始数据的有效特征属性值;(b) 划分算法,即根据数据的特征属性值将数据集划分成若干类;(c) 有效性评价以及结果解释,即对聚类的结果进行评价并给出面对实际背景的合理解释.聚类算法的研究主要指上面聚类过程中第 2 层次的研究,其充分利用了机器学习、神经网络、数据库、多元统计、图论等多学科的理论和技术,成为图像分割、数据压缩、信息检索等越来越多的理论和应用领域中的关键课题[1−6].

聚类算法的主要任务是将数据集按照“类内相似、类间相异”的原则进行分组.简洁直观的方法就是每一类数据都与该类中的某个代表点或核心点充分接近,同时远离其他类的代表点或核心点,这种代表点或核心点一般称为聚类中心.典型的算法包括平均链接[1,2]、*K*-means 算法[7]和 *K*-medoids 算法[8].平均链接和 *K*-means 算法都是以平均值来表示该类,相比之下, *K*-means 算法具有更低的计算复杂度.由于平均值易受到异常值或噪声的影响,因此 *K*-medoids 算法在每类中选取与其他点距离之和最小的真实数据作为聚类中心(如图 1 所示).这样虽然改善了对噪声点和异常值的鲁棒性,但也增加了运算复杂度,不适合大规模数据.Kaufman 和 Ng 等人引入随机采样的技巧,相继提出 Clara[8]和 Clarans 算法[9],克服了 *K*-medoids 算法复杂度较高的问题,使其适用于大规模数据.

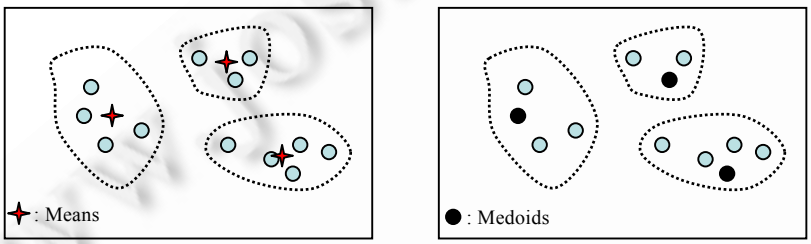


Fig.1 Cluster centers in *K*-means and *K*-medoids

图 1 *K*-means 和 *K*-medoids 算法中聚类中心的选取

那么,如同 *K*-means 算法和 *K*-medoids 算法,当我们需要把某数据集分为 *K* 类时,为每一类选择一个聚类中心是否合适呢?若数据形态呈现为凸的球状或云团状,且数据的各类之间没有明显的交叉重叠,同时没有严重的噪声污染时,合适的聚类中心的确可以准确表达数据的结构,获得满意的聚类结果.但是,当数据呈现非凸的形状、重叠或噪声较严重时,只靠一个点(聚类中心)难以准确表述聚类的结构.为此,研究者开始考虑利用多个点表述聚类的结构,比较典型的方法包括 CURE 算法[10]以及基于密度的算法[11,12].CURE 算法是利用固定 *m* 个点表示一个类,这 *m* 个点称为代表点.代表点的引入,使得 CURE 算法能够得到分离较好的非球状和任意非凸形状的聚类.代表点的选择遵循尽可能在该类中分散开的原则(如图 2 左图所示,圆圈和三角为 CURE 算法中两类的代表点),同时使其按一定比率地向每类的均值向量(类中心)收缩(即引入一个收缩常数)以减少噪声的影响.另外,在 CURE 算法中,类与类之间的距离定义为它们代表点之间距离的最小值,这类似于单链接算法,但收缩常数的引入使得 CURE 算法对噪声有一定的鲁棒性.然而,CURE 算法中收缩常数往往难以确定:太小(接近 0),则受噪声或野值的影响仍然很大;太大(接近 1),则收缩到类均值附近,这样只能获得类似球状的聚类(因为当收缩常数为 1 时, CURE 算法即等价于 *K* 均值算法).

基于密度的聚类算法如 DBSCAN[11],NBC(neighborhood-based clustering)[12]是将数据集中密度较大的点(稠密点)视为数据集的代表点(如图 2 右图所示,圆圈为 DBSCAN 算法中的稠密点).DBSCAN 算法将固定半径邻域中包含数据个数超过定值的点视为稠密点,然后建立从稠密点出发的凝聚聚类,直到某类中所有稠密点的邻域中不包含新稠密点为止.相比前面的算法,DBSCAN 不需要事先确定聚类个数,但算法对其两个参数(邻域半径以及邻域内数据个数的阈值)非常敏感,往往需要反复调整才能获得好的聚类结果.NBC 算法利用类似 DBSCAN 的凝聚聚类思想,但在判断稠密点时,根据每个数据自己 *k* 邻域和反 *k* 邻域内包含数据个数的对比情况进行判断.这样,NBC 只需要一个参数,即最近邻个数 *k*,提高了算法效率.

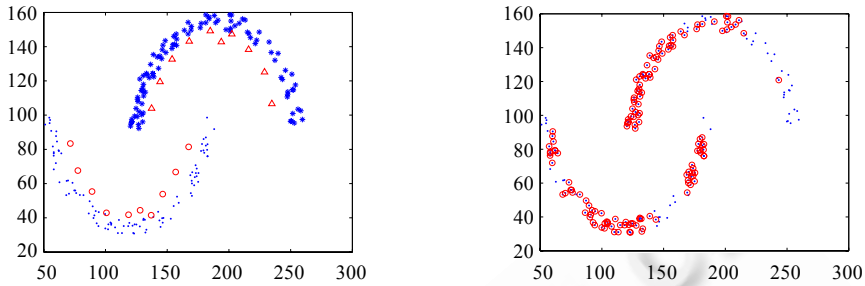


Fig.2 Representative or dense points in CURE and DBSCAN

图 2 CURE 和 DBSCAN 算法中代表点或稠密点的选取

此外,作为 DBSCAN 方法对于网络数据聚类的一个推广应用,SCAN 算法[13]建立了针对网络数据特有的结构化准则,将数据区分为 3 种不同的结构点,即毂、质心(centriod)和野值;并使用质心作为数据集的代表点.SDTC(structured directed-tree clustering)则是针对一般数据集的结构化聚类算法[14],具体做法是在数据邻域计算时引入多项式核,使其质心点的确定少受噪声干扰,从而提高算法的鲁棒性.然而,虽然利用数据集合中的稠密点作为代表点极大地增强了表达数据结构的能力,但是稠密点的判断一直是所有基于密度的聚类算法的关键问题.一方面易受到参数变化、噪声野值的影响;另一方面,对于密度不均匀和存在聚类重叠的数据集往往效果不佳.

通过上面的算法分析可以发现,聚类中心一直是表述聚类结构的重要方式.这里的聚类中心包括 *K*-means 算法中的均值、*K*-medoids 算法中的 medoid、CURE 算法中的代表点、DBSCAN 和 NBC 算法中的稠密点、SCAN 和 SDTC 算法中的质心等等.另外,聚类中心的数目也是从一个(e.g. *K*-means)到固定的 *m* 个(e.g. CURE)以及到不确定的几个(e.g. DBSCAN),这使得方法能够表述的聚类结构也从分离的球状凸形进展到拉长非凸的以及任意混合的多种形状.但是需要指出的是,面对如今日渐复杂的数据,现有这些基于聚类中心的算法仍然显得力不从心.

一方面,现在许多新兴产业和应用领域产生的数据更多地呈现出“类间相似、类内相异”的聚类结构,往往包含大量噪声或野值,聚类之间存在明显重叠,分离性不好.对此,很多算法如 CURE,DBSCAN,即使对所有参数进行不断调整也难以获得满意聚类结果;另一方面,随着互联网的日益普及,数据获取更加便利,数据规模也随之扩大.对此,大部分算法经常需要过多地进行迭代,甚至迭代次数有时根本难以确定.例如,CURE 算法需要(N–k)次凝聚过程(其中,N 为采样数据集规模,k 为聚类个数),而 *K*-means 或 *K*-medoids 的迭代次数依赖于初始中心的选择,这给实际应用带来极大不方便.

为提高聚类算法的效率,近年来,众多研究者纷纷引入 Multi-level 聚类的思想方法.例如,针对复杂的图划分问题,Karypis 等人提出 METIS 方法[15,16],但是该方法类似于 *K*-way 最小割,只倾向于找到规模大小极为接近的聚类.为此,Dhillon 等人对图粗划分后的分层结果,利用加权的核 *K*-means 方法对其进行分层、细化,实现了比 METIS 更高效的聚类过程.2006 年,针对自然图像,Sharon 等人基于代数多网格算子理论,提出一种基于加权凝聚的分层分割算法(segmentation by weighted aggregation,简称 SWA)[18].具体地,输入图像的像素点集作为最底层的顶点,SWA 逐层一次性地凝聚前一层中所有顶点的一半左右作为后一层的顶点,直至“目标分割块”凝聚成功出现为止;逐层凝聚时,选择与原始图像中其他像素最相似的顶点保留下来,这些顶点即类似于图像中某个局部邻域的中心点.这样,SWA 通过从底向上的金字塔形的多层凝聚过程快速有效地实现了图像中目标的分割和提取.

受此启发,本文将提出一种基于多层聚类中心(称为核心集)的凝聚聚类算法 MulCA(multilevel core-sets based aggregation),主要包括两方面:

1) 每一层数据向其自身核心集进行粗化凝聚.具体来讲,计算每一层数据间的相似性得到该层数据集的首要核心点,进而获得该层数据集α比例(α<1)的核心集,核心集中的每个点相当于一个局部的聚

类中心;进一步地,下层(即前一层)的核心集自动成为上层(即后一层)的数据集,每层数据集规模按α比例迭代减少,直至最顶层数据集个数等于聚类个数时,粗化迭代过程终止(如图 3 第 1 行——多层核心集的粗化选取所示);

2) 顶层核心集逐层往下进行凝聚细化.具体而言,虽然顶层的每个数据点都代表了某个聚类,但聚类结构并不是只由这一个点表述,而是通过定义上下层核心集数据间的相似性逐层往下对各层核心集进行凝聚细化,最终形成该聚类(如图 3 第 2 行——逐层的凝聚细化所示).

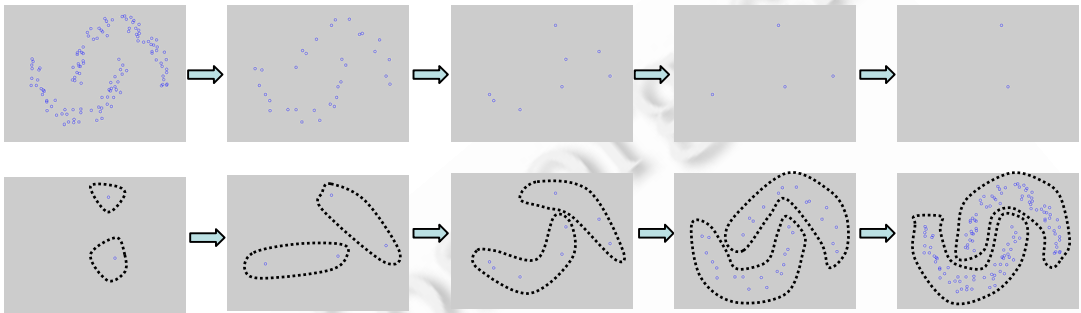


Fig.3 Two phases in MulCA

图 3 MulCA 算法的两个过程

相比前面介绍的基于代表点的聚类算法,MulCA 算法不是利用某层(即数据集本身最细那一层)某个或某些静态的核心点表述一个聚类,而是通过多层核心集的凝聚演化动态表述一个聚类;而与其他 Multi-level 聚类算法不同的是,当下层的核心集成为上层的数据集之后,上层数据集中的每一个点都对应着下层中的一个类,凝聚了该类中其他点的信息,这样的逐层凝聚使得 MulCA 算法对于噪声以及密度不均等问题具有更好的鲁棒性.可以说,动态表述和逐层凝聚是 MulCA 相对于其他代表点和 Multi-level 聚类算法的最大优势.进一步,MulCA 算法中每层核心集数目按α比例迅速减少,有效控制了整个算法的凝聚演化复杂度.

此外,MulCA 算法可以推广到大规模数据集的聚类.对于较小规模的数据集,MulCA 算法的底层数据一般取为数据集本身.对于较大规模的数据集,则可以用原始数据集的ε(ε接近于 0)比例核心集作为底层数据,而通过引入随机采样的方法可以处理更大规模的数据集.

本文第 1 节详细介绍 MulCA 算法.第 2 节为 MulCA 算法的实验分析以及与几种经典的基于代表点聚类算法的比较.第 3 节将算法推广到大规模数据集聚类.最后一节总结全文.

1 MulCA:基于多层核心集凝聚的聚类算法

1.1 基于高斯核的相似性与α-核心集

度量或相似性的定义是几乎所有聚类算法中必不可少的部分,常用的度量或相似性包括距离度量(欧式距离、最大/最小值距离、马氏距离、明氏距离等)、核相似性、余弦相似性、密度相似性等.

高斯核相似性是一种常用的相似性定义方式,其对噪声和边界具有一定的鲁棒性[4,19,20],因此,本文利用高斯核定义数据间的相似性.

定义 1(数据间的相似性). x_i和 x_j的相似性定义为

$$\mu_\sigma(x_i,x_j)=\mathrm{e}^{-\frac{\|x_i-x_j\|^2}{\sigma^2}},$$

其中,$\sigma=\sigma_0 d$,d 为数据集的直径.

首先,利用相似性定义数据集中的首要核心点:

定义 2(首要核心点 x^*). 对于任意数据集 X,如果点 x^*满足:

$$x^* = \arg\max_{x\in X} \sum_{\substack{y\in X \\ y\neq x}} \mu_\sigma(x,y),$$

则称 x^*为集合 X 的首要核心点.

有了首要核心点,自然可以定义次要核心点、第三核心点…一般地,可以定义数据集的α比例核心集:

定义 3(α-核心集). 设 X 为任意数据集,记 x_1^* 为其首要核心点, x_2^* 为集合 $X\setminus\{x_1^*\}$ 的首要核心点.一般地,记 X 的前 k 个核心点分别为 $x_1^*,x_2^*,...,x_k^*$,令 $X^{(k)} = X\setminus\{x_1^*,x_2^*,...,x_k^*\}$,定义 X 的第 k+1 个核心点为 $X^{(k)}$的首要核心点,即

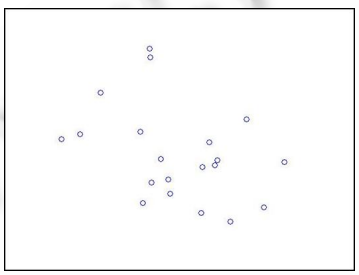
$$x_{k+1}^* = \arg\max_{x\in X^{(k)}} \sum_{\substack{y\in X^{(k)} \\ y\neq x}} \mu_\sigma(x,y).$$

X 的α-核心集定义为

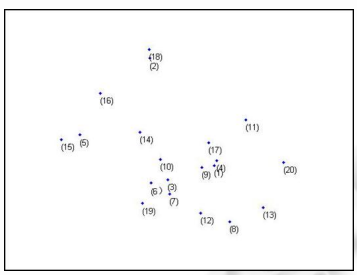
$$X_\alpha = \{x_1^*, x_2^*,...,x_k^*\},$$

其中,k=[$\alpha\cdot|X|$](|X|表示 X 中元素的个数,[·]为取整函数);α为 0~1 之间的实数,用于控制 X 的核心集的规模.

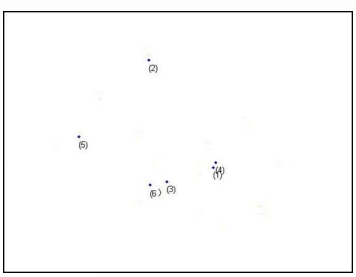
通过随机生成 20 个点,图 4 给出了 $x_1^*,x_2^*,...,x_{20}^*$ 的分布情况(如图 4(b)所示)以及不同α对应的α-核心集示例图(如图 4(c)~图 4(e)所示).可以看出,当α较小时,数据集的α-核心集往往是一些稠密点的集合.注意到第 k+1 个核心点为数据集移去前 k 个核心点后的首要核心点,因此α-核心集中的点往往是彼此分散开的,并没有完全集中在数据集的稠密区域,它们类似于一些分散在数据集中的局部的聚类中心.



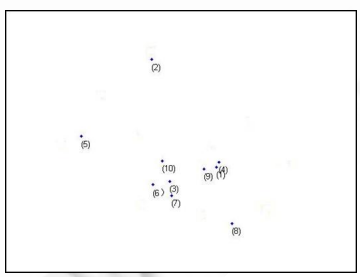
(a) 原始数据图



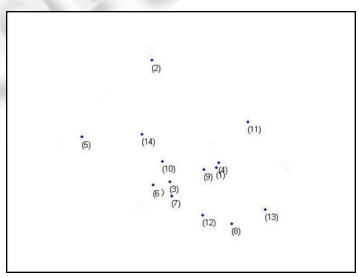
(b) 数字表示按定义 3 得到的核心点的顺序



(c) α=0.3 时的α-核心集



(d) α=0.5 时的α-核心集



(e) α=0.7 时的α-核心集

Fig.4 Determination of Corel-points and α-Corel-sets

图 4 核心点的选取以及α-核心集

1.2 基于α-核心集的凝聚分类

由定义 2 和定义 3 可知,α-核心集体现了原数据集中元素基于相似性的相对重要性,其中的数据点往往是局部的稠密点,类似于一些分散在数据集中的局部的聚类中心,因此可以通过 X 的α-核心集将 X 凝聚为若干类.显然,与某个核心点相似性较大的 X 中的点应该凝聚在该核心点所在的聚类中.基于这种思想,给出 X 到 X 的α-核心集的凝聚矩阵.

定义 4(凝聚矩阵). 设 X 为任意数据集,由集合 X 到其α-核心集 X_α的凝聚矩阵定义为

$$P_{X\text{-}X_\alpha}(m,n)=\begin{cases}1, & m\in X_\alpha, m=n\\ 0, & m,n\in X_\alpha, m\neq n\\ \dfrac{\mu_\sigma(m,n)}{\sum_{t\in X_\alpha}\mu_\sigma(m,t)}, & m\in X\setminus X_\alpha, n\in X_\alpha\end{cases}.$$

显然, $P_{X\text{-}X_\alpha}$ 是$|X|\times|X_\alpha|$的矩阵.凝聚矩阵体现的是集合 X 与其α-核心集 X_α中数据间的相关的程度.当 $m=n$ 时,对应的凝聚矩阵中的元素值为 1,这表示数据集中每个数据都与其自身最相关;当 $m\neq n$ 且同时为核心集中的点时,对应的凝聚矩阵中的元素值为 0,这表示核心集中的点不直接相互影响.而当 m 不是核心点、n 为核心点时,对应的凝聚矩阵中的元素值表示非核心点依附于核心点的程度.注意到,其正比于两点的相似性.因此,X 到其α-核心集 X_α的凝聚矩阵相当于将 X 划分为$|X_\alpha|$类的模糊划分矩阵.

由α-核心集 X_α借助凝聚矩阵 $P_{X\text{-}X_\alpha}$ 可以将原数据集凝聚为$|X_\alpha|$类,下面给出凝聚算法:

凝聚算法(aggregation algorithm).

Input X, $X_\alpha=\{x_1^*,x_2^*,,...,x_k^*\}$;

Initialize $C_i=\{x_i^*\}, i=1,2,...,k$.

For t=1:n {

$m=\arg\max_{1\leqslant j\leqslant k} P_{X-X_\alpha}(t,j)$

C_m=$C_m\cup\{x_t\}$

}

Output C=$\{C_1,C_2,\ldots,C_k\}$.

图 5 给出了图 4(a)数据集向其α-核心集(α=0.3)图 4(c)凝聚的过程,将该数据集聚为 6 类.

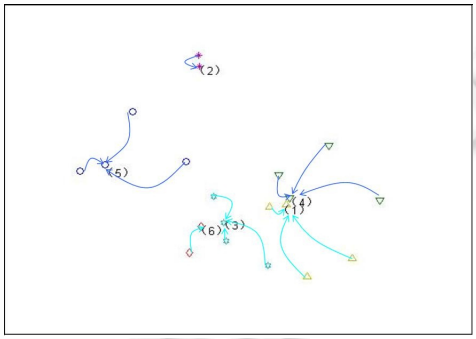


Fig.5 Clustering results of MulCA for the data in Fig.4 (a) by the α-Corel-set in Fig.4 (c)

图 5 利用图 4(c)中的α-核心集得到的图 4(a)的聚类结果

虽然 X 的α-核心集 X_α可以将 X 划分为$|X_\alpha|$类,但是未必能获得理想的聚类结果.特别地,若 X 由具有不同密度的部分组成,根据高斯核相似性的定义,X 排在前面的几个核心点将集中在密度较大的部分,这样得到的聚类结果往往是不合理的(如图 6 所示,从左到右依次为原图(有两部分组成,左侧密度较大,右侧密度较小)、原图的前两个核心点(三角形)标记图以及数据集向两个核心点凝聚得到的聚类结果图).究其原因在于,α-核心集 X_α的规模太小时,无法体现数据集的整体信息.

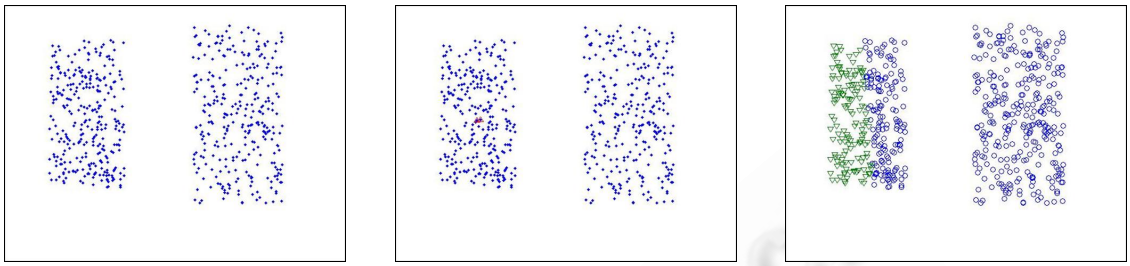


Fig.6 Faliure case of MulCA based on the one-level α-Corel-set

图 6 基于单层α-核心集的 MulCA 算法聚类失败的例子

1.3 多层α-核心集

对于较复杂的数据集,只通过 X 的α-核心集 X_α可以将 X 凝聚为$|X_\alpha|$类,陷入两难境地——$|X_\alpha|$过小,易出现图 6 所示的情况,核心点集中在密度较大区域;$|X_\alpha|$过大,则会将数据集凝聚为太多的类,无法找到完整的聚类结构.鉴于此,本文引入多层α-核心集,通过逐层凝聚的方法,试图找到复杂数据集中完整的聚类结构.所谓多层α-核心集模型,即将前一层数据集 X 的α-核心集 X_α作为后一层的数据集,继续计算其α-核心集,直到最顶层α-核心集的个数等于预设的聚类个数为止.

当计算 X_α的α-核心集时,如何得到 X_α的数据之间的相似性呢?显然,直接用高斯核相似性函数(定义 1)无法体现前后两层的凝聚关系.注意到,X_α中的每个数据点相当于 X 中的一个类,这样,X_α的数据之间的相似性相当于 X 中的两个类之间的相似性.由于凝聚矩阵 $P_{X\text{-}X_\alpha}$ 是将 X 划分为$|X_\alpha|$类的模糊划分矩阵,自然可以利用 X 中每个点与聚类中心(即 X_α中的点)的隶属关系结合 X 中两点间的相似性定义两类之间的相似性(即聚类中心—— X_α 中的点之间的相似性).图 7 给出了 X_α中的点 k 和 l 相似性关系建立的示意图(k,l 表示 X_α中的两个点,i 和 j 表示 X 中任意的两个点,$\mu_\sigma^{(X_\alpha)}(k,l)$ 为 k,l 的相似性),下面给出 X_α中的相似性关系的定义:

定义 5(X 的α-核心集 X_α中数据间的相似性). 设任意集合 X,X_α为 X 的α-核心集, $P_{X\text{-}X_\alpha}$ 为 X 到 X_α的凝聚矩阵,则 X_α的加权相似性矩阵定义为

$$\mu_\sigma^{(X_\alpha)} = P'_{X\text{-}X_\alpha}\mu_\sigma^{(X)}P_{X\text{-}X_\alpha},$$

其中, $\mu_\sigma^{(X)}$ 表示数据集 X 上的相似性矩阵, $\mu_\sigma^{(X_\alpha)}$ 表示其α-核心集 X_α上的相似性矩阵.

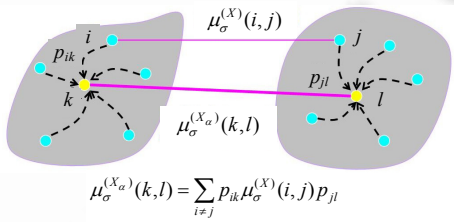


Fig.7 Similarity determination between data-points in X_α

图 7 X 的α-核心集 X_α中数据间相似性的定义示意图

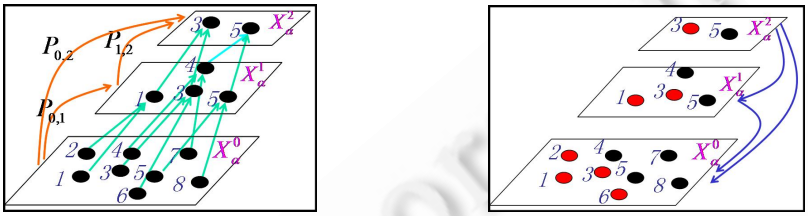
归纳地,由定义 5 可以给出原数据集 X 的任意一层α-核心集的相似性.设 X_α^t 为 X 的第 t 层α-核心集,则 X_α^{t+1} 上的相似性为

$$\mu_\sigma^{(X_\alpha^{t+1})} = P'_{t,t+1}\mu_\sigma^{(X_\alpha^t)}P_{t,t+1},$$

其中,$P_{t,t+1}$ 是集合 X_α^t 到其α-核心集 X_α^{t+1} 的凝聚矩阵 $(X_\alpha^0 = X)$.

1.4 多层α-核心集凝聚算法(MulCA)

建立多层α-核心集模型后,通过逐层凝聚,得到多层α-核心集凝聚聚类算法(MulCA).如图 8 所示,MulCA 算法主要包括两个步骤:形成金字塔形多层核心集的过程以及通过逐层凝聚将数据集分类的过程.图 8(a)表示形成金字塔形多层核心集的过程,图 8(b)表示通过逐层凝聚将数据集分类的过程.最终将该数据集凝聚为两类,分别为 1,2,3,6(灰色)和 4,5,7,8(黑色).



(a) MulCA 算法步骤 1 (b) MulCA 算法步骤 2

Fig.8 A general algorithmic illustration of MulCA

图 8 MulCA 算法示意图

MulCA 算法的两个步骤具体为:

MulCA 算法步骤 1. 计算数据集 X 的各层α-核心集直到最顶层的元素个数等于预设聚类个数,形成金字塔形多层核心集.具体步骤如下:

Input X, K (number of clusters);

Initialize $X_\alpha^0 = X, m = \left\lceil \log_\alpha \frac{K}{n} \right\rceil$.

Compute similiar matrix $\mu_\sigma^{X_\alpha^0}$ of X_α^0 from definition 1

Compute α-core set X_α^t of X_α^0 from definition 3

For t=1:m−1 {

Compute $P_{t,t+1}$, $\mu_\sigma^{(X_\alpha^{t+1})}$ from definition 4, definition 5

Compute α-core set X_α^{t+1} of X_α^t from definition 3

}

Output $X_\alpha^1,...,X_\alpha^m,P_{0,1},...,P_{m-1,m}$.

MulCA 算法步骤 2. 通过逐层凝聚,将原数据集划分为 K=| X_α^m |类.具体步骤如下:

Input X, $P_{0,1},\ldots,P_{m-1,m}$;

Compute $P_{0,m}$=$P_{0,1}$·$P_{1,2}$·…·$P_{m-1,m}$.

Initialize $C_i = \{x_i^*\}, i = 1,2,...,k$.

For t=1:n {

$i = \arg\max_{1 \le j \le k} P_{0,m}(t,j)$

C_i=C_i∪{x_t}

}

Output cluster C={C_1,C_2,…,C_m}.

算法中,X_α^t 表示数据集 X 的第 t 层α-核心集,K 表示需要聚类的个数,α-核心集的层数为 $\left\lceil \log_\alpha \frac{K}{n} \right\rceil$.在每层的凝聚过程中,用 X_α^t 中的每一个数据代表着凝聚到与该点为一类的所有点,并同时将其代表的所有点传递到下

一次凝聚过程中去.因此在 MulCA-step2 中,使用了凝聚矩阵的逐层乘积得到最终的模糊划分矩阵 $P_{0,m}$.

然而,对于所输入的 K 值,不一定存在合适的 m 值使得 $\underbrace{[\alpha[\alpha...[\alpha\,|\,X\,|]]]}_{m\text{次取整}}=K$.为此,给定一个大于 $\left[\frac{K}{\alpha}\right]$ 的正整数 M,在算法 MulCA-step1 中,当 $|X_\alpha^{t-1}|\geqslant M$ 时,不改变算法,继续求 X_α^{t-1} 的 α-核心集 X_α^t;当 $|X_\alpha^{t-1}|<M$ 时,则每层计算核心集,其规模不是该层数据集的 α 倍,而仅比该层数据集的规模小 1,直到等于 K 为止.这样,就保证了可以达到预定的聚类个数 K.简单计算可知,核心集的层数不超过 $\left[\log_\alpha\frac{M}{n}\right]+M$.

1.5 算法的复杂度

首先给出说明算法正确性的定理:

定理 1. 对于任意的有限数据集 X、正整数 K 以及介于 0 和 1 之间的实数 α,算法在有限步终止.

定理显然可由上一节关于核心集的层数不超过 $\left[\log_\alpha\frac{M}{n}\right]+M$ 的结论得到(其中,n 为数据集的规模,M 可取为大于 $\left[\frac{K}{\alpha}\right]$ 的正整数).

在该算法中,α 为 0~1 之间的实数,M 一般小于 20.一般来说,α 不能太大也不能太小:当 α 接近于 0 的时候,凝聚算法中每层得到的 α-核心集仅仅取了最核心的部分点,对于大部分数据集一般得不到正确的聚类结果,如图 2 所示;当 α 接近于 1 的时候,每层得到的 α-核心集将把大部分点作为核心点提取出来,这显然过于强调数据集的局部信息,从而不能充分反映出它的整体结构.因此实际算法中,一般将 α 限定在 0.3~0.7 之间.

MulCA 算法的计算复杂度

第 1.3 节将 MulCA 算法分成两个步骤:得到多层核心集以及逐层凝聚.在相似性矩阵已知的前提下,主要的计算量集中在:

(1) 相似性的求和与排序,求每一层的核心集;

(2) 相邻两层凝聚矩阵的计算;

(3) 核心集相似性矩阵的计算;

(4) 由凝聚矩阵得到数据集聚类结果.

对于步骤(1),由于核心集的规模呈现等比数列递减,所有层核心集包含核心点的总数约为 $\frac{\alpha n}{1-\alpha}$,这样,步骤(1)的计算复杂度不超过 $O\left(\frac{\alpha}{1-\alpha}n^2\ln n\right)$;对于步骤(2),每一次计算凝聚矩阵的复杂度为上下两层核心集个数的乘积,这样,所有凝聚矩阵的计算复杂度不超过 $O\left(\frac{\alpha n^2}{1-\alpha^2}\right)$;对于步骤(3),根据定义 5,由凝聚矩阵计算核心集的相似性矩阵涉及到矩阵乘法,因此,所有核心集相似性矩阵的计算复杂度不超过 $O\left(\frac{\alpha n^c}{1-\alpha^2}\right)$(其中,$c$ 为大于 2 但接近于 2 的常数[21]);对于步骤(4),聚类结果只需要求出凝聚矩阵每一行的最大值,因此,计算复杂度不超过 $O(nK\ln K)$.可见,步骤(3)的计算复杂度是最高的,因此,MulCA 算法的计算复杂度为 $O\left(\frac{\alpha}{1-\alpha^2}n^c\right)$($c$ 为大于 2 但接近于 2 的常数).由相似性矩阵的定义 1,相似性矩阵具有一定的稀疏性,因此,整个算法的计算复杂度近似于数据集相似性矩阵的计算复杂度 $O(n^2)$,这对于中小规模的数据集是可以接受的.

2 实验比较及评价

在本节中,将 MulCA 算法与具有单一代表点的 K-means 算法、平均链接算法以及多个代表点的 DBSCAN 算法、基本 CURE 算法进行比较,实验数据集包括具有各种形状、不同重叠和噪声的人工数据集以及部分真实

数据集,其中,第 2.1 节是对任意形状数据集的实验,第 2.2 节是对不同重叠程度数据集的实验,第 2.3 节是对加入不同数量噪声数据集的实验,第 2.4 节是对部分真实数据集的实验,第 2.5 节则对 MulCA 算法中的参数进行了讨论.在第 2.1 节~第 2.4 节的实验中,算法的参数设置如下:

- *K*-means 和平均链接算法:参数只有一个,即聚类数目,在所有实验中,该参数均取为数据集中真实的聚类个数;
- DBSCAN 算法:参数分别为定义稠密点的邻域半径 r 和个数 k(在具体实验中,k 一般取为 10 左右,r 根据数据集的直径决定);
- 基本 CURE 算法:参数分别为代表点数目 m 和收缩常数 c(在具体实验中,m 一般取在 6~10 之间,c 一般取在 0.2~0.8 之间);
- MulCA 算法:共有两个参数,其取值范围和默认取值见表 1.参数σ_0为高斯核的核宽参数,所有使用高斯核相似性的算法都会涉及到此参数.MulCA 算法对σ_0的选取与其他算法相似,默认取值为 0.05(对于维数较高的数据集,如 USPS,一般取在 0.1~0.2 之间).参数α为 MulCA 算法特有的参数,其控制了核心点的选取比例.当α较小时,下一层核心集的规模较小,算法速度较快;当α较大时,下一层核心集规模较大,算法速度较慢.但是,由于α太小时易出现图 6 所示的情况,因此在实验中,α的取值一般限定在 0.3~0.7 之间,默认取值为 0.5.

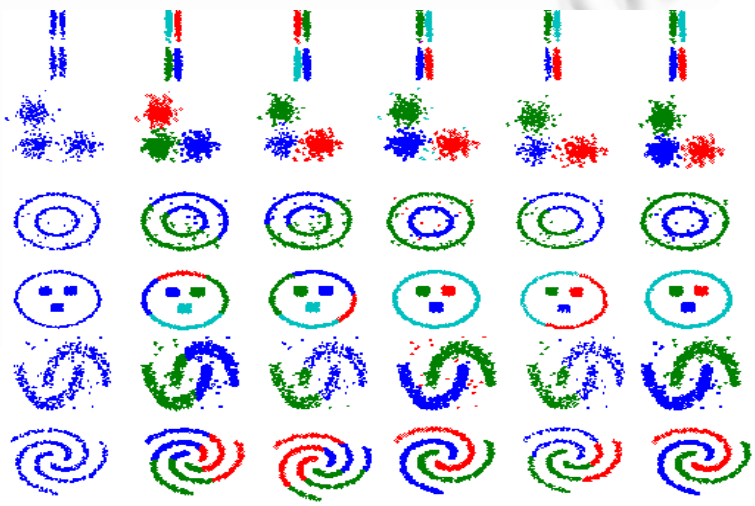
Table 1 Parameter setting in MulCA

表 1 MulCA 算法的参数设置

MulCA 算法参数	参数意义	取值范围	默认取值
α	控制核心点的选取比例	0.3~0.7	0.5
σ_0	高斯核的核宽	0.02~0.20	0.05

2.1 任意形状的人工数据集实验

好的聚类算法能够聚类任意形状的数据,本节将本文算法与 4 种经典的聚类算法(*K*-means、平均链接、DBSCAN、CURE)对细长型、云团状、流形状等各种形状的数据集合进行聚类实验.6 种不同形状的人工数据集均使用不同类型的高斯分布随机生成(图 9 中第 1 列),样本点数从上到下分别为 400,600,610,400,660,675(其中,第 3 个和第 5 个数据集分别包含 50 个和 60 个分布在数据集空间域的随机噪声).



(1) 原图 (2) *K*-means (3) 平均链接 (4) DBSCAN (5) CURE (6) MulCA

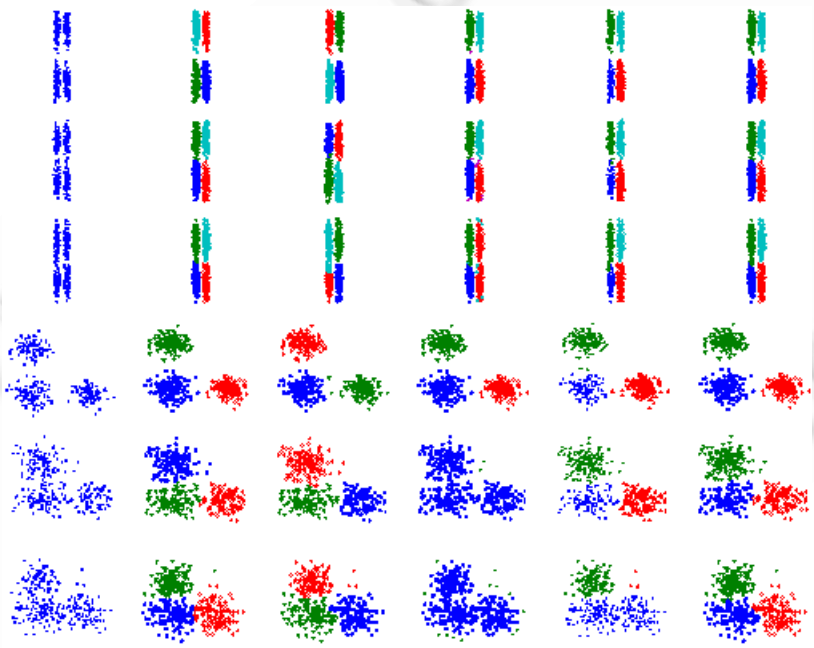
Fig.9 Comparison of clustering methods over six data-sets with different shaped-clusters

图 9 不同形状数据聚类的结果比较

实验结果(图 9 中第 2 列~第 6 列)表明,DBSCAN 和 MulCA 算法能够聚类任意形状的数据集,而其余算法对非凸数据及聚类结果不理想(DBSCAN 算法的参数 r 和 k 从上到下依次为 4,10;0.33,10;0.5,15;0.6,10;17,10;0.9,10;CURE 算法的参数 m 和 c 从上到下依次为 6,0.2;10,0.6;7,0.7;10,0.2;7,0.6;10,0.8;MulCA 算法的两个参数 α 和 σ_0 均取为默认值 0.5 和 0.05).

2.2 不同重叠程度的人工数据集实验

当数据集逐渐重叠时,聚类的难度变大.在本节,对两种人工数据集(图 9 的前两行分别对应的细长型数据和团状数据),考虑重叠程度变大时,不同算法的聚类结果如图 10 所示,前 3 行为重叠程度不断增加的细长型数据集,各包含 400 个数据点,后 3 行为重叠程度不断增加的云团状数据集,各包含 390 个数据点.可以发现,*K*-means、平均链接和 MulCA 算法呈现出对重叠较好的鲁棒性,而 DBSCAN 与 CURE 算法对于重叠增大时的聚类效果不理想(DBSCAN 算法的参数 r 和 k 从上到下依次为 4,10;33,9;29,9;0.4,10;0.5,12;0.5,10;CURE 算法的参数 m 和 c 从上到下依次为 6,0.2;6,0.6;6,0.6;6,0.6;6,0.6;6,0.6;MulCA 算法的两个参数 α 和 σ_0 均取为默认值 0.5 和 0.05).



(1) 原图 (2) *K*-means (3) 平均链接 (4) DBSCAN (5) CURE (6) MulCA

Fig.10 Comparison of clustering methods over the data-sets with different degrees of overlap

图 10 不同重叠程度的聚类结果比较

2.3 加入不同个噪声点的人工数据集实验

为了检验 MulCA 算法对噪声的鲁棒性,针对两种非凸人工数据加入逐渐增多的噪声点.图 11 左边为原图(包含 300 个数据点)中分别加入 20,30,40 个随机噪声,右边为原图(包含 540 个数据点)中分别加入 30,50,70 个随机噪声(第 1 行为原图,第 2 行为 MulCA 算法结果;MulCA 算法的两个参数 α 和 σ_0 从左到右依次取为 0.6,0.04;0.5,0.05;0.5,0.05;0.5,0.05;0.5,0.05;0.6,0.04).

可以看出,MulCA 算法对噪声具有较好的鲁棒性.

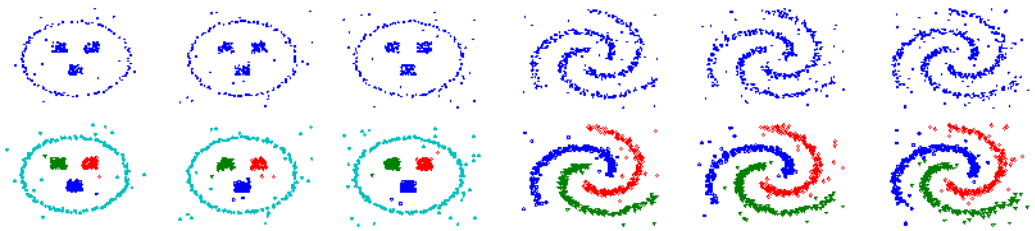


Fig.11 Clustering results over the noisy data set

图 11 噪声点逐渐增加时的聚类结果

2.4 真实数据集实验

为了进一步说明 MulCA 算法的有效性,本节将在部分真实数据集上进行聚类实验.选取的真实数据集分别来源于 USPS(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,本文的实验只取其中的测试样本)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,本文的实验选取其中的 wine 和 iris 两个数据集),实验所用数据集的具体情况如下:

- usps (0,1):USPS 测试样本中选出数字 0 和 1 的样本,共包含 623 个数据;
- usps (0,2):USPS 测试样本中选出数字 0 和 2 的样本,共包含 557 个数据;
- usps (4,9):USPS 测试样本中选出数字 4 和 9 的样本,共包含 377 个数据;
- usps (5,8):USPS 测试样本中选出数字 5 和 8 的样本,共包含 326 个数据;
- usps (0,1,5):USPS 测试样本中选出数字 0,1,5 的样本,共包含 783 个数据;
- wine:来源于 UCI 数据库,包含 178 个 13 维数据,分为 3 类;
- iris:来源于 UCI 数据库,包含 150 个 4 维数据,分为 3 类.

在实验中,MulCA 算法的参数(α,σ_0)和 DBSCAN 算法的参数(r,k)分别标注在准确率的后面括号中,K-means 和 Average-linkage 的参数都取为实际的聚类数目.从表 2 中可以看出,与其他算法相比,MulCA 算法显示了极好的聚类结果.

Table 2 Clustering results of real data-sets

表 2 真实数据聚类结果

数据集	Accuracy			
	MulCA (%)	k-means (%)	Average-Linkage (%)	DBSCAN (%)
usps (0,1)	**99.84** (0.3,0.08)	98.56	57.46	94.22 (9.4,12)
usps (0,2)	**94.61** (0.4,0.03)	82.05	64.63	66.61 (10.3,12)
usps (4,9)	75.60 (0.5,0.2)	**78.78**	52.79	44.03 (8.6,11)
usps (5,8)	**88.96** (0.4,0.1)	84.36	51.23	37.12 (9.6,11)
usps (0,1,5)	**81.48** (0.6,0.04)	79.80	45.72	70.37 (8.6,11)
wine	**71.91** (0.3,0.05)	70.22	38.76	53.37 (51,7)
iris	**96.00** (0.3,0.05)	89.33	68.67	73.33 (0.5,11)

2.5 MulCA算法的参数实验与讨论

图 12 给出了α取不同值时,6 个数据集的平均聚类准确率.可以看出,不同的α取值对大部分数据集的聚类效果相似.

图 13 给出了σ_0取不同值时,6 个数据集的平均聚类准确率.可以看出,当σ_0取值在 0.05 附近时,对大部分数据集的聚类效果较好(6 个数据集分别为第 2.1 节图 9 的第 1 行、第 3 行、第 6 行数据集以及 Iris,wine 和 usps (0,1)).

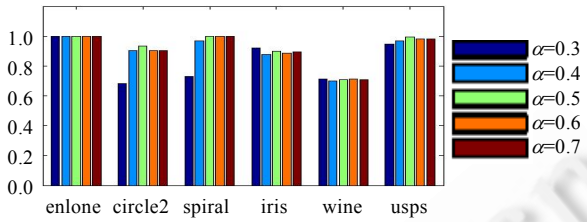


Fig.12 Average clustering accuracy varing with α

图 12 α取不同值时,6 个数据集的平均聚类准确率

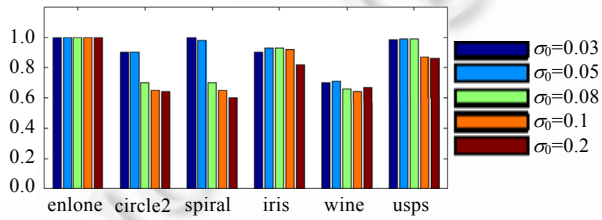


Fig.13 Average clustering accuracy with different σ_0

图 13 σ_0取不同值时,6 个数据集的平均聚类准确率

3 应用于大规模数据集

3.1 应用于大规模数据集的算法框架

对于大规模数据集,由于 n 很大,根据第 1.4 节得到的 MulCA 算法的计算复杂度为 $O\left(\frac{\alpha}{1-\alpha}n^2\ln n\right)$,因此,MulCA 算法难以直接应用.为此,可以利用以下步骤得到适用于大规模数据的 MulCA 算法:

(a) 先得到 X 的较小规模的子集 X_0;

(b) 在 X_0 上使用 MulCA 算法进行聚类;

(c) 将聚类结果从 X_0 推广到 X(本文中,通过 X 与 X_0 的相似性矩阵与步骤(b)得到的 X_0 的模糊划分矩阵相乘得到原始数据集 X 的模糊划分矩阵).

如图 14 所示,数据集 X 共包含 35 000 个点,在步骤(a)中,分别通过 3 种方法(直接求ε-核心集、随机采样以及基于随机采样求ε-核心集)得到包含 700 个点的子集 X_0;在步骤(b)中,对子集 X_0 使用 MulCA 算法进行聚类;在步骤(c)中,将聚类结果分别推广到整个数据集.

若 X_0 的数据个数为 n_0,则步骤(b)与步骤(c)的计算复杂度接近 $O\left(\frac{\alpha}{1-\alpha}n_0^2\right)$ 和 $O(nn_0)$,二者之和约为 $O(n_0n)$,远远低于直接在 X 上直接用 MulCA 算法的复杂度.于是,步骤(a)成为关键.要得到 X 的较小规模的子集 X_0,需考虑以下 3 种方法:

(1) 取远远小于 1(接近 0)的正数ε,以 X 的ε-核心集 X_ε作为 X_0.在已知数据集相似性矩阵的前提下,计算 X_ε 的复杂度约为 $O(\varepsilon n^2)$,与步骤(b)、步骤(c)之和的复杂度相仿;但若数据集的相似性矩阵未知,则需要先计算相似性矩阵,复杂度为 $O(n^2)$.由此可见,这种方法对于成对数据、网络数据等已知相似性矩阵的数据集是可行的,但是对于相似性矩阵未知的数据集,并不能根本地降低算法的复杂度(图 14 中的第 1 行).

(2) 如同已有的一些适用于大规模数据集的聚类方法(Clara[8]、Clarans 算法[9]、CURE 算法[10]等)引入随机采样的技巧,直接以随机采样得到的数据集作为 X_0,这样几乎不需要增加任何的计算量,但是一次随机采样往往具有很大的不确实性,易受到噪声等的干扰,无法确保聚类结果(图 14 中的第 2 行,由于噪声的影响,聚类结果不

理想).

(3) 由于上面的两个方法都有明显的缺陷,考虑一种新的得到 X_0 的方法:首先对 X 进行随机采样得到较小规模的数据集为 X'(设其个数为 n');然后计算 X 中每个数据与 X'中每个数据的相似性的和,将相似性的和进行排序,按照定义 3 得到 X 的ε-核心集作为 X_0.这样就不需要计算数据集 X 中两两点之间的相似性,计算相似性的复杂度减少为 $O(n'n)$,求 X的ε-核心集 X_ε的复杂度仍为 $O(\varepsilon n^2)$.这种基于随机采样计算ε-核心集(random sampling based ε-core set computation,简称为 RBC)的方法可以有效弥补上面两种方法的缺陷(图 14 中的第 3 行,先随机采样了包含 3 000 点的数据集 X',然后得到了 X_0,相比第 2 行直接随机采样得到的 X_0 更能体现数据集的结构,因而在只有第 1 行复杂度 10%的情况下,得到了同样的聚类结果).

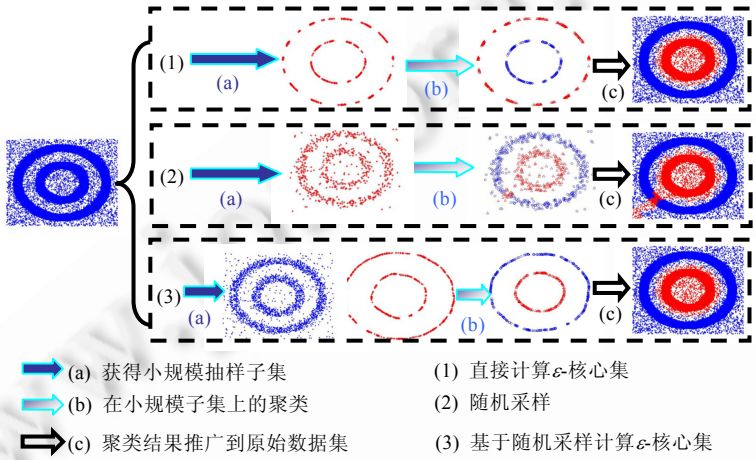


Fig.14 Framework of MulCA on the large scale data

图 14 MulCA 算法在大规模数据集的应用框架

3.2 实验分析

本节将通过实验说明基于 RBC 方法的 MulCA 算法在大规模数据集聚类中的有效性(若无特别说明,MulCA 算法中的参数均为第 2.5 节中的默认取值).

首先考虑 RBC 方法(基于随机采样计算ε-核心集)对噪声的鲁棒性.选取原始数据集为包含 15 万个点的同心圆环,分别加入 5 万、10 万、15 万和 20 万个均匀分布的随机噪声,实验结果如图 15 所示(4 列分别为 3 000 点随机采样集、RBC 得到的 800 点核心集、核心集的聚类结果以及原始数据集的聚类结果).可以看出,基于 RBC 的 MulCA 算法对逐渐增加噪声的大规模数据集具有很好的鲁棒性.

基于 RBC 方法首先进行随机采样,然后计算原始数据集与随机采样集的相似性矩阵,由此得到原始数据集的ε-核心集.计算核心点主要利用数据之间的相似性,因此,高斯核的核宽 σ_0 会影响核心集的选取.下面,通过实验说明基于 RBC 方法的 MulCA 算法对相似性矩阵中核宽参数 σ_0 的鲁棒性.

取实验数据集为包含 15 万个点的同心圆环,并加入 2 万个平均分布的随机噪声.图 16 和图 17 分别显示了使用随机采样的 MulCA 算法和使用 RBC 方法的 MulCA 算法的聚类结果.显然,RBC 方法对参数具有更好的鲁棒性.图 16 中,内外两个同心圆环共包含 150 000 个点,加入 20 000 个均匀分布的噪声点,利用随机采样的方法进行 MulCA 聚类(第 1 行分别为原始数据集和随机采样集,第 2 行为不同参数得到的采样集聚类结果,第 3 行为最终得到的数据集的聚类结果).图 17 中,内外两个同心圆环共包含 150 000 个点,加入 20 000 个均匀分布的噪声点,利用 RBC 方法进行 MulCA 聚类(第 1 行分别为原始数据集、随机采样集和核心集,第 2 行为不同参数得到的核心集聚类结果,第 3 行为最终得到的数据集的聚类结果).

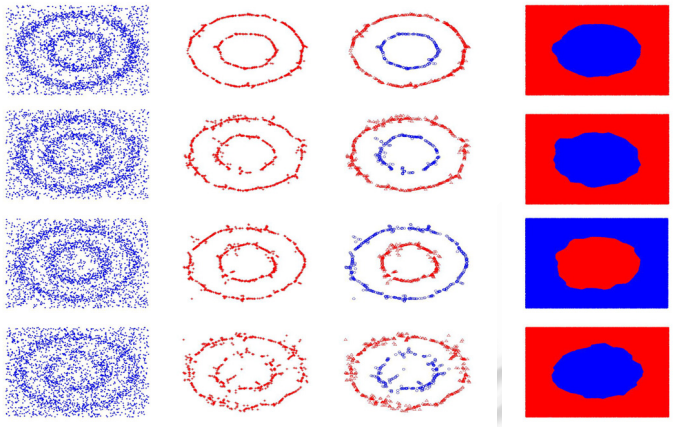


Fig.15 Results of RBC-based MulCA on the data with more and more noise

图 15 不断增加噪声时,使用随机采样计算ε-核心集(RBC)的 MulCA 算法效果

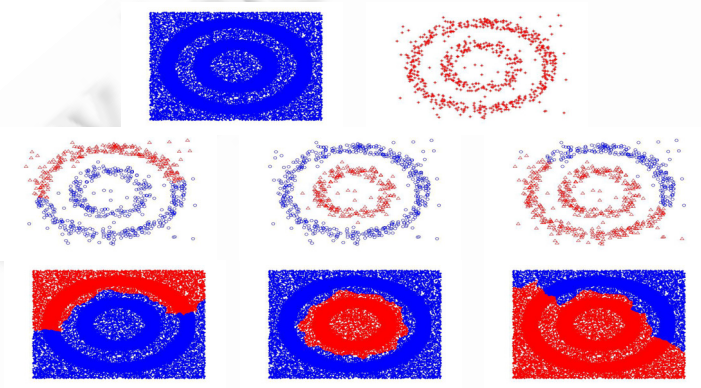


Fig.16 Clustering results of MulCA by random sampling

图 16 使用随机采样的 MulCA 算法的聚类结果

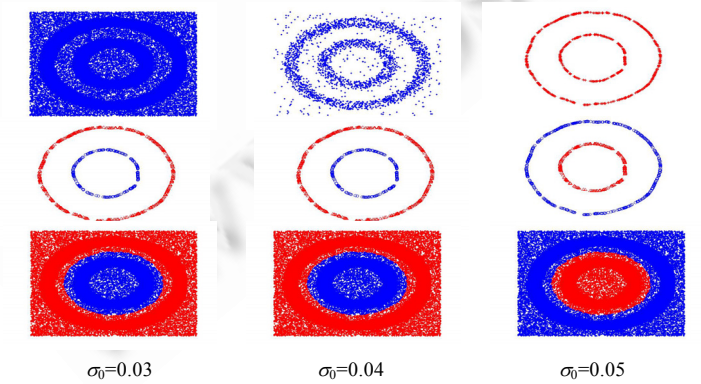


Fig.17 Clustering results of RBC-based MulCA

图 17 使用 RBC 方法的 MulCA 算法的聚类结果

4 总 结

利用聚类中心进行聚类是一种重要的聚类方法,产生了诸如 *K*-means、平均链接、Clara、CURE、DBSCAN 等许多经典的聚类算法.其发展过程是从单一的聚类中心(如平均链接,*K*-means,*K*-medoids,Clara,Clarans 等算法)到固定数目的多个聚类中心(如 CURE 等算法),再到不确定数目的聚类中心(很多算法中称为稠密点或质心点,如 DBSCAN,NBC,SCAN,SDTC 等算法).聚类中心数量的增多,的确使得聚类中心所表述的聚类结构从球状、凸状到任意形状,但是也带来参数增加、对数据的重叠或噪声敏感等缺点.

为此,本文提出一种基于多层聚类中心(称为核心集)的凝聚聚类算法.该算法创新性地使用了多层核心集凝聚动态表述聚类结构,使得每一层数据集向其核心集凝聚.同时,上层的核心集自动成为下层的数据集.随着每层核心集规模按α比例迅速减少,控制了凝聚过程的迭代次数.在人工数据集和实际数据集的聚类实验中,将 MulCA 算法与几种经典聚类算法进行比较,并对 MulCA 算法中的参数进行了实验分析.

此外,本文还引入了基于随机采样计算ε-核心集的技巧,将 MulCA 算法成功应用于大规模数据集.通过大规模数据集的聚类实验,说明了本文提出的 RBC 方法在聚类结果、噪声与参数的鲁棒性等方面都优于直接随机采样的方法.

今后的进一步工作将着眼于 MulCA 聚类算法框架的推广应用,使其能适用于更多种数据型态的复杂聚类问题.

致谢 衷心感谢匿名审稿人对本文改进提出的宝贵意见.

References:

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

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

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

[4] Shi JB, Malik J. Normalized cuts and image segmentation. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2000,22(8):888−905. [doi: 10.1109/34.868688]

[5] Pal NR, Pal SK. A review on image segmentation techniques. Pattern Recognition, 1993,26(9):1277−1294. [doi: 10.1016/0031-3203(93)90135-J]

[6] Datta R, Joshi D, Li J, Wang JZ. Image retrieval: Ideas, influences, and trends of the new age. ACM Computing Surveys, 2008,40(2):1−60. [doi: 10.1145/1348246.1348248]

[7] 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.

[8] 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]

[9] 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]

[10] 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]

[11] 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.

[12] 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]

[13] Xu XW, Yuruk N, Feng ZD, Schweiger TAJ. SCAN: A structural clustering algorithm for networks. In: Proc. of the 13th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. New York: ACM Press, 2007. 824−833. [doi: 10.1145/1281192.1281280]

[14] Ding JD, Ma RN, Chen SC. Polynomial kernel based structural clustering algorithm by building directed trees. Ruanjian Xuebao/Journal of Software, 2008,19(12):3147−3160 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/19/3147.htm [doi: 10.3724/SP.J.1001.2008.03147]

[15] Karypis G, Kumar V. A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM Journal on Scientific Computing, 1999,20(1):359−392. [doi: 10.1137/S1064827595287997]

[16] Abou-Rjeili A, Karypis G. Multilevel algorithm for partitioning power-law graphs. In: Proc. of the IEEE Int'l Parallel & Distributed Processing Symp. (IPDPS). 2006. 1−10. [doi: 10.1109/fIPDPS.2006.1639360]

[17] Dhillon I, Guan Y, Kulis B. A fast kernel-based multilevel algorithm for graph clustering. In: Proc. of the 11th ACM Conf. on Knowledge Discovery and Data Mining (KDD 2005). New York: ACM Press, 2005. 629−634. [doi: 10.1145/1081870.1081948]

[18] Sharon E, Galun M, Sharon D, Basri R, Brandt A. Hierarchy and adaptivity in segmenting visual scenes. Nature, 2006,442(7104):810−813. [doi: 10.1038/nature04977]

[19] Zhang D, Chen S. Clustering incomplete data using kernel-based fuzzy *C*-means algorithm. Neural Processing Letters, 2003,18:155−162. [doi: 10.1023/B:NEPL.0000011135.19145.1b]

[20] Camastra F, Verri A. A novel kernel method for clustering. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2005,27(5):801−805. [doi: 10.1109/TPAMI.2005.88]

[21] Cohn H, Umans C. A group theoretic approach to fast matrix multiplication. In: Proc. of the 44th Annual Symp. on Foundations of Computer Science. Cambridge: IEEE Computer Society, 2003. 438−449. [doi: 10.1109/SFCS.2003.1238217]

附中文参考文献:

[14] 丁军娣,马儒宁,陈松灿.基于多项式核的结构化有向树数据聚类算法.软件学报,2008,19(12):3147−3160. http://www.jos.org.cn/1000-9825/19/3147.htm [doi: 10.3724/SP.J.1001.2008.03147]

马儒宁(1976－),男,山东济宁人,博士,副教授,主要研究领域为应用数学,模式识别.
E-mail: mrning@nuaa.edu.cn

王秀丽(1987－),女,硕士生,主要研究领域为聚类算法.
E-mail: wang.xiuli.2008@163.com

丁军娣(1978－),女,博士,讲师,CCF 会员,主要研究领域为模式识别,计算机视觉.
E-mail: dingjundi@nuaa.edu.cn