文章编号：2095-2163(2022)05-0023-09 中图分类号：TP391 文献标志码：A

一种基于改进型 Chameleon 算法的宿舍分配方法

顾唐杰，秦 波，蒋小菲
（贵州大学 大数据与信息工程学院，贵阳 550025）

摘 要：本文基于 Chameleon 算法对高维度 R 型问题进行聚类分析，提出加权共享-变色龙算法（KSNN-Chameleon）用于改进学生宿舍分配问题。KSNN-Chameleon 算法首先将学生的宿舍集通过 K 近邻算法稀疏化，然后采用近邻加权方式处理已稀疏化的数据集得到加权近邻图；接着对加权近邻图通过洪水覆盖法（flood-fill）和点间共享近邻相似度（SNN）进行图划分；最后 KSNN-Chameleon 算法采用第一截断法来快速确认数据集的分簇是否符合要求，反复划分与合并获得最终的聚类结果。实验结果证明，KSNN-Chameleon 算法不仅在面对 R 型高维度聚类问题时仍能保证较好的稳定性与精度，且与传统 Chameleon 算法相比，KSNN-Chameleon 的聚类精度提升了 20.88%，聚类时间提升了 2.73%。

关键词：Chameleon 算法；高维 R 型聚类分析；共享最近邻；宿舍分配

A dormitory allocation method based on improved Chameleon algorithm

GU Tangjie, QIN Bo, JIANG Xiaofei
(College of Big Data and Information Engineering, Guizhou University, Guiyang 550025, China)

【Abstract】 In order to solve the intelligent assignment problem of the student dormitory, the paper proposes the K-Shared Nearest Neighbour-Chameleon algorithm (KSNN-Chameleon) based on the traditional Chameleon algorithm for clustering analysis of high-dimensional R-type problems. The KSNN-Chameleon algorithm first processes sparsely the obtained student dormitory set by the K-nearest neighbor algorithm, and then the weighted nearest neighbor graph is obtained by processing the sparsed data set using nearest neighbor weighting; the weighted nearest neighbor graph is later divided into graphs by flood-fill method and the data are merged using the shared nearest neighbor (SNN) similarity method; finally, the KSNN-Chameleon algorithm uses the first truncation method to quickly confirm the subclustering of the dataset, and the final clustering results are obtained by iterative division and merging. The experimental results demonstrate that the KSNN-Chameleon algorithm not only ensures better stability and accuracy when facing the R-type high-dimensional clustering problem, but also improves the clustering accuracy of KSNN-Chameleon by 20.88% and the clustering time by 2.73% compared with the traditional Chameleon algorithm.

【Key words】 Chameleon algorithm; high-dimensional R-type clustering analysis; shared nearest neighbor; dormitory allocation

0 引 言

当代年轻人拥有多样化的性格，这也将导致部分学生在人际交往过程中会出现各种各样的问题，在这些问题中室友往往担任着非常重要的角色。良好的宿舍氛围不但促进了学习进步，也提高了学生的生活质量，为大学生心理健康起到正面积极作用。然而近几年关于同寝室犯罪的报道却已开始映入眼帘，抛开极端情况不谈，在普通宿舍中也不时会出现因室友性格不同而导致的宿舍“小团体”现象或校园霸凌问题。根据新华网一项针对大学生舍友关系的调查显示，42.28%的学生与舍友曾经发生矛盾；与舍友发生矛盾时，仅有 47.81%的学生会选择“积极沟通”[1]。显然对于部分学生的矛盾很难通过教育、引导途径化解。

在此背景下智能宿舍分配方法孕育而生，2016 年，上海大学的高校公寓“私人定制”计划采用了网上选室友计划[2]。而在国外，学生也可以通过网络选择室友[3]。为进一步满足学生宿舍需求，本文通过调研学生共性，对学生中性格习性相近或互补的学生进行聚类，并对 Chameleon 算法做出改进得到一种全新算法。

Chameleon 算法是由 CURE 和 ROCK 算法演变而来，兼顾了接近度和内部互联度，在二维数据的聚类中取得了非常好的效果[4]。目前，Chameleon 算法也在不断的发展中出现过比较典型的改进。例如龙真真等人[5]提出了 M-Chameleon 算法，改进后不但减少了聚类过程中所需的参数，并能更加客观地

基金项目：贵州大学省级本科教学内容和课程体系改革项目(2021003)。

作者简介：顾唐杰(1997-)，男，硕士研究生，主要研究方向：软件开发、智能算法处理、软件服务；秦 波(1996-)，男，硕士研究生，主要研究方向：智慧教育与智能系统；蒋小菲(1988-)，女，博士，副教授，硕士生导师，主要研究方向：物联网及信号处理。

收稿日期：2021-12-06

反映聚类情况。宫峰勋等人的《基于 DPC 算法与模块密度的改进 Chameleon 算法》中,在传统算法的基础上于图划分阶段利用密度峰值算法使稀疏图的划分更为准确,并在后续聚类过程中采用集群数据点相似性的函数获得最终的聚类结果[6]。

结合以上算法优点,本文通过对 Chameleon 算法改进得到 KSNN - Chameleon 算法。KSNN - Chameleon 算法在针对学生宿舍分配问题时,能够通过学生之间共性的差异进行智能宿舍分配。在查阅相关资料和社会调查后,本文按照学生的兴趣爱好、月生活水平、学习习惯、作息时间、嗜好等[7],对每一位学生的特点进行划分并通过 Matlab 软件进行分析推测出适合每一位学生的室友。

1 算法理论分析

本算法是在 Chameleon 算法的基础上进行改进和提炼的新型 KSNN-Chameleon 算法。本节将介绍 KSNN-Chameleon 算法的理论依据,以及对传统算法中各个阶段的改进。

1.1 Chameleon 算法

聚类分析中的层次分析法可以笼统地分为:由整体分裂的方法和由多点凝聚的方法。这取决于层次分解当中分解的顺序,而 Chameleon 算法采用动态建模的方式进行聚类[8]。Chameleon 算法流程如图 1 所示。由图 1 可知,首先需要对数据项进行稀疏化,然后通过相关的图划分算法对稀疏图进行划分,从而形成多个相对连接性较高的初始子簇。最后使用凝聚层次聚类技术,重复合并相似性高的簇。即 Chameleon 算法是先对整体进行分裂,又采用凝聚的方法进行二次聚类的聚类算法。Chameleon 算法最大的优点在于,既考虑了每个簇的相对互连度 $RI(C_i,C_j)$,又考虑到簇的相对邻近度 $RC(C_i,C_j)$,通过二者相结合的相似度函数计算数据点之间的距离。

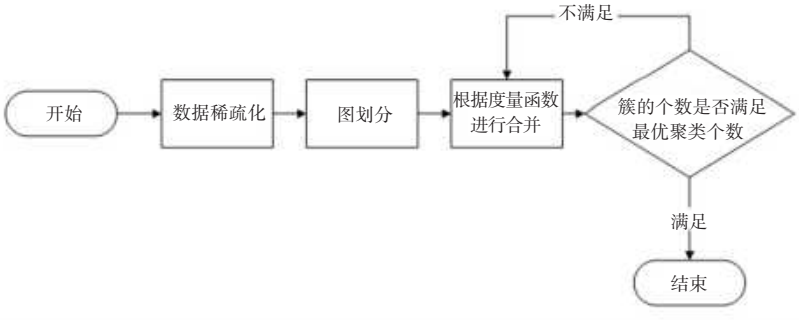

图 1 Chameleon 算法流程

Fig. 1 Chameleon algorithm flow chart

定义 1 相对互连度 $RI(C_i,C_j)$ 2 个簇 C_i 和 C_j 之间的绝对互连度关于 C_i 和 C_j 的内部互连度的规范化,即:

$$RI(C_i,C_j)=\frac{|EC_{\{C_i,C_j\}}|}{\frac{1}{2}(|EC_{C_i}|+|EC_{C_j}|)} \tag{1}$$

定义 2 相对邻近度 $RC(C_i,C_j)$ 2 个簇 C_i 和 C_j 之间的绝对接近度关于 C_i 和 C_j 的内部接近度的规范化,公式如下:

$$RC(C_i,C_j)=\frac{\bar{S}_{EC_{\{C_i,C_j\}}}}{\frac{|C_i|}{|C_i|+|C_j|}\bar{S}_{EC_{C_i}}+\frac{|C_i|}{|C_i|+|C_j|}\bar{S}_{EC_{C_j}}} \tag{2}$$

其中,$\bar{S}_{EC_{\{C_i,C_j\}}}$ 是连接顶点 C_i 和顶点 C_j 的边的平均权重,$\bar{S}_{EC_{C_i}}$ 或 $\bar{S}_{EC_{C_j}}$ 是最小二分簇 C_i 或 C_j 的边的平均权重。

定义 3 相似性度量函数 相对互连度和相对互连度的乘积,即:

$$Sim(C_i,C_j)=RI(C_i,C_j)\times RC(C_i,C_j)^{\alpha} \tag{3}$$

其中,α 是指定参数,设定两值的权重。例如,当 $\alpha=1$,代表 2 个度量标准权重相同;如果 $\alpha>1$,则 Chameleon 更偏重于相对接近度;反之,当 $\alpha<1$ 时,则偏重相对近邻度[9]。

1.2 改进稀疏化

Chameleon 算法的第一个阶段(稀疏化过程)是通过 K 最近邻法求出数据点的 K 最近邻图的过程,得到的 K 近邻图被称为数据集的稀疏图。此方法计算数据点关系时采用的是欧式距离法,对于偏重于距离分析的 Q 型聚类问题,这种方式可以有效地

将数据点进行分类。然而，当遇到利用数据相关程度进行聚类的 R 型聚类问题时，这种聚类方法的适用性就会有所下降。因此，为了更好地利用稀疏图进行图划分，KSNN-Chameleon 算法在稀疏化得到 K 近邻图后，将 K 近邻图转换为加权近邻图。

1.2.1　利用 KD 树减少总体聚类时间

由于本算法需要对 K 近邻图进行转换，聚类时间也随之增加。KSNN-Chameleon 在 K 近邻算法的基础上利用 KD-tree 原理对其进行改进，大大减少了聚类所需要的时间。

定义 4　K 最近邻法　根据目标点 x 的距离度量，在输入数据集 T 中找出与 x 最近的 k 个点，涵盖着 k 个点的领域，记为 $N_k(x)$，若 $N_k(x)$ 中某一个簇类 C_i 的出现的次数最多，则判定目标点 x 也属于 C_i。

定义 5　KD-tree　是一种对 K 维空间中的实例点进行存储以便对树形树状结构进行快速检索。是空间二分树的一种特殊情况[10]。

1.2.2　加权近邻图

传统 K 近邻图仅仅只能体现出数据点之间的位置关系和距离关系，而面对 R 型聚类问题时，利用欧式距离描述数据点之间的位置关系是不太准确的。为更好地解决 R 型聚类中，稀疏图依赖距离的问题，本文将得到的稀疏图，通过近邻度权重的方式进行改良从而得到加权近邻图。利用加权近邻图进行图划分可以更有效地避免 R 型聚类问题中依赖距离因素的问题[11]。

定义 6　两点之间的近邻度　公式如下：

$$A(i,j)=a_{ij}/r \tag{4}$$

其中，r 是分簇个数，a_{ij} 是点 i 和 j 被分为同一个簇的次数。若 $A(i,j)\geqslant 0.5$，则 i 和 j 为彼此的近邻点[12]。

定义 7　近邻权重　设 $\Gamma(i)$，$\Gamma(j)$ 是数据点 i 和 j 的近邻列表，$|\Gamma(i)|$ 表示 i 的所有近邻数，$|\Gamma(j)|$ 表示 j 的所有近邻数，则数据点 i 和 j 的近邻权重可以定义为以下公式：

$$\sigma_{ij}=|\Gamma(i)\cap\Gamma(j)| \tag{5}$$

交集个数表明，若两者越相似，则所拥有的共同近邻点就越多，权重越大。根据以上定义得到算法 1，用以得到加权近邻图[13]。

算法 1　Weighted nearest neighbor graph

输入：含 n 个数据点的数据集 $X=\{x_1,x_2,\cdots,x_n\}$

输出：加权近邻图

```
1: Int N      //X 中数据点的个数
2: For (i = 1; i <= N; i ++)
3:     For (j = 1; j <= N; j ++)
4:         A(i, j) = associations(i, j);
           //计算两点近邻度
5:     End for
6: End for
7: If ( A(i, j) >= 0.5) then
8:     do i ∈ Γ(j), j ∈ Γ(i)
9: End if
10: If (i ∈ Γ(i) ‖ j ∈ Γ(j)) then
11:    σ_ij = Γ(i) ∩ Γ(j)
       // 计算两点间的权值
12: End If
13: add_weighed_edge(i, j, σ_ij)
    //得出加权近邻图
```

1.3　改进图划分

Chameleon 算法的第二阶段在得到稀疏图后会直接采用 hMetis 划分算法对获得的稀疏矩阵进行图划分。hMetis 是一种多层次的分割方法，在处理粗糙图形信息上具有良好的效果并且聚类的时间更短[14]。但考虑到 hMetis 算法不开放源代码，并且 hMetis 算法是一种采用最小可能性切割边缘进行图划分的方法，有时会产生距离很远却聚为一类的问题，所以在处理一些信息时达不到理想的聚类效果。为了更好地实现图划分，KSNN-Chameleon 算法采用 flood-fill 法结合递归二分法[15]，代替原有的 hMetis 算法，对得到的加权近邻图进行图划分。详情见算法 2。

算法 2　flood-fill

输入：graph

输出：多连接子簇

```
1: while i do
2:     For i ∈ neighbor(node) do
3:         If i is null then
4:             Cluster = new_Cluster()
               //创建新空间存放数据点
5:             mark_connected_subgraph
               (graph, i, cluster)
6:         End If
7:     End for
8: End while
9: mark_connected_subgraph(graph, i, cluster)
10: Mark(node, cluster) // 标记簇中的点
```

11: For j in *neighbor*(*node*) do

12: If j is *null* then

13: *mark_connected_subgraph*(*graph*, j, *cluster*)// 建立多连接簇

14: End If

15: End for

定义 8 洪水覆盖法(flood-fill) 给区域内某一个点 a 上色,并对所有相邻的点依次涂上 a 的颜色,不断填充此区域,直到区域内的所有点都附上颜色。

1.4 合并阶段

本算法在 Chameleon 聚类的第三个阶段合并时主要针对 2 个部分进行改良。一方面传统算法在针对二维数据集时,采用的相似性度量函数可以有效地将图划分后的各个数据簇进行分析和合并,但是在面对三维以上的数据集时,其聚类效果会随着维度的升高而下降。为了在高维 R 型聚类问题中,得到更加精准的聚类簇,KSNN-Chameleon 算法采用共享最近邻相似度的方法进行改进,从而避免高维情况下的维数灾难。另一方面,传统算法中需要反复计算目标数据集的最优聚类个数,这一过程消耗大量的时间,而 KSNN-Chameleon 算法采用第一截断法的方式,对该过程进行简化,可得到数据集聚类的最佳簇数。

1.4.1 共享最近邻相似度

为了解决高维度问题,本文引入共享最近邻相似度代替传统相似性度量,由于共享最近邻相似性度量在空间中主要反映的是点的局部结构,因此对空间的维度并不敏感,所以可作为一种相对合适的度量方法。共享最近邻相似度可以理解为:对于 2 个不同的数据点而言,如果彼此的相似点中,存在着大量重合的部分,那么即可判定这 2 个数据点也相似。如图 2 所示,点 i 和 j 都有 8 个最近邻居,其中有 4 个点是 i 和 j 都共有的,称为共享邻居,所以,点 i 和点 j 之间的最近邻相似度为 4[16]。

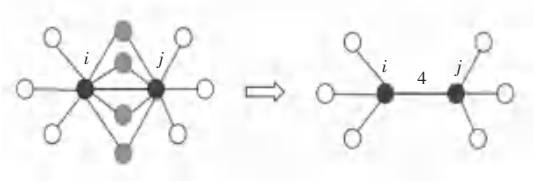

图 2 共享邻居展示图

Fig. 2 Shared nearest neighbour diagram

定义 9 共享邻居 在数据集 X 中存在任意 2 点 i 和 j,将点 i 的 k 个近邻集合设为 $\Gamma(i)$,同理将 j 的集合设为 $\Gamma(j)$,则点 i 和 j 的的公共邻居集,定义如下:

$$SNN(i,j)=\Gamma(i)\cap\Gamma(j) \tag{6}$$

定义 10 最近邻相似度(Shared Nearest Neighbour,SNN) 对于数据集 X 的任意数据点 i 和 j,其共享最近邻相似度可定义为:

$$SNN=\begin{cases}\dfrac{|SNN(i,j)|^2}{\sum_{p\in SNN(i,j)} d_{ip}+d_{jp}} & i,j\in SNN(i,j)\\ 0 & \text{其它}\end{cases} \tag{7}$$

1.4.2 第一截断法

为了更有效地合并子簇并减少获取数据的时间,在子簇合并的过程中使用“第一截断法”[17]。该方法的原理是:通过 2 个簇合并的位置视为“第一个大跃变”,此时 2 个簇理论上最不相似,因此只需找到第一个大跃变位置,如此就能找到最适合的分簇。可将合并后的层级视为自下而上的树状图,将树状图从中间划分为上、下两个部分,不断切割直至找到第一大跃变点作为合并图的最优截断数据,该数据即为数据集的最优分簇,具体代码详见算法 3。

算法 3 First Jump Cutoff Algorithm

输入:合并图 S

输出:合并图的最优截断高度

1: Procedure *FIRSTJUMP*(*initMultiplier*, *factor*)

2: *avgJump* = *compute_avgJump*()

3: *Multp* = *initMultiplier*()

4: while *multp* > 0 do

5: *res* = *find_biggerJump*(*multp* * *avg*)

6: If *res.flag then*

7: return *res.height*

8: else

9: *multp* = *multp*/*factor*

10: End if

11: End while

12: *find_biggerJump*(*jump*)

13: *result* = *newList*()

14: For all *level* do //*level* ∈ *Dendrogram*

15: *levelWidth* = *level.next.height* − *level.height*

16: If *LevelWidth* > *jump* then

17：　　*result.flag = true*
18：　　*res.height = level.height*
19：　Return *result*
20：　End If
21：End for
22：*res.flag = false*
23：Return *result*

2　KSNN-Chameleon 实现

KSNN-Chameleon 算法与传统算法均分为 3 个阶段，传统的 Chameleon 算法在针对二维 Q 型算法时，能够得到良好的聚类效果。但其算法在面向如“宿舍分配”等多维 R 型聚类问题时，难以维持二维情况下的精准聚类。因此本算法在传统算法的各个阶段都进行了不同程度的改良，使得改良后的算法在面对高维度的 R 型聚类问题时也能取得良好的聚类效果。

2.1　构建加权近邻图

KSNN-Chameleon 算法以加权近邻图的方式代替原有的 K 近邻图作为数据集的稀疏图。首先采用 KD 树和 K 近邻算法相结合的方式，获取 K 近邻图，然后，根据公式(3)计算出每个点的相似度，再通过公式(4)和公式(5)计算出每个点近邻点集，得到由近邻点集构造出的加权近邻图。具体步骤如下所示。

输入：　初始数据集 X，分类参数 k

输出：　加权近邻图(稀疏图)

Step 1　构建 KD 树，通过输入参数 k 对相似度矩阵进行初始化。

Step 2　从 KD 树的树根开始不断遍历，将所有数据划分在不同的区域当中。

Step 3　根据数据集 X 中的数据，利用定义 3 中的公式(3)算出每个点间的相似度，构建相似度矩阵。

Step 4　在 KD 树的每个区域中取前 K 个最相似的值分类更新相似度矩阵，得到 K-最近邻相似矩阵。

Step 5　利用算法 1，分别求出不同 k 值时得到相似矩阵情况，根据定义 6、7 计算数据点间的近邻权重，得到加权近邻图。

2.2　加权近邻图的划分与合并

在 KSNN-Chameleon 算法的第二阶段和第三阶段中，对加权近邻图进行图划分时，首先利用递归二分法和洪水覆盖法(flood-fill)对稀疏图进行图划分如算法 2 所示，然后采用定义 10 中的公式(7)计算出共享相似度，对图划分后得到的子簇进行自下而上的层次聚类，此时可以得到数据集的合并树状图。此后采用算法 3 中的第一截跳法求出适合于数据集的最佳聚类数，反复合并划分图直到满足最终簇数完成聚类，具体步骤如下所示。

输入：　加权近邻图，聚类簇数 r

输出：　r 个簇的聚类

Step 1　将加权近邻图带入算法 2 中，利用 flood-fill 方法对加权近邻图进行图划分，得到分簇。

Step 2　结合定义 10 中公式(7)求出各个子簇之间的共享相似度，推算子簇之间的相似关系。

Step 3　通过共享相似度对数据集进行自下而上的层次聚类，并将聚类结果视为树状图 S。

Step 4　将树状图 S 带入算法 3 中，利用第一截断法求出聚类簇数。

Step 5　将得到的簇数与聚类簇数 r 对比，若不同则返回 Step 2；若相同，则此时的聚类结果即为最终的聚类结果。

结合 KSNN-Chameleon 算法三个阶段所绘制的大致流程如图 3 所示。

图 3　KSNN-Chameleon 算法总流程

Fig. 3　Flow chart of KSNN-Chameleon algorithm

3　实验数据分析

3.1　KSNN-Chameleon 评估

为验证 KSNN-Chameleon 算法的可行性，本文采用 Matlab 测试数据集中具有代表性的 4 个测试数据集、即 Aggregation、Iris、Seeds、Wine 数据集对改良算法进行测试，并通过几种聚类评估的基准验证 KSNN-Chameleon 算法对于高维度的数据集具有良

好的聚类效果。在与传统算法进行对比后得到的聚　　类结果见表 1。

表 1　不同维度下 2 种算法聚类对比

Tab. 1　Clustering comparison of two algorithms in different dimensions

数据集	样本数	维度数	Chameleon 聚类精度	KSNN-Chameleon 聚类精度	KSNN-C 聚类时间/Chameleon 时间
Aggregation	788	2	0.895	0.901	1.03
Iris	150	4	0.784	0.921	0.99
Seeds	210	7	0.789	0.936	0.98
Wine	178	13	0.733	0.944	0.99

KSNN-Chameleon 算法针对 Aggregation 二维数据集聚类后的结果如图 4 所示,在面向二维数据集时,KSNN-Chameleon 算法与传统算法得出的结果图基本一致。图 5 为 Iris(四维数据集)在三维空间中的展示图,通过对比后证明在高维聚类的情况中,KSNN-Chameleon 算法比传统 Chameleon 算法的聚类更加完善,其中聚类精度平均提升了 20.88%,聚类时间平均提升了 2.73%。实验证明在面对 R 型高维度聚类问题时,KSNN-Chameleon 算法仍然能具有较好的稳定性与聚类精度。

本实验对 289 名在校学生进行调研,并将数据整理为表,见表 2。结合图 4、图 5 和表 2 中数据可看出,对于二维数据集而言,KSNN-Chameleon 算法的聚类精度没有明显地高于传统 Chameleon 算法,仅仅是略高于传统型。但当数据集的维度变大时,传统 Chameleon 算法的聚类效果明显下降,而改进型则有了显著的提升,即 KSNN-Chameleon 算法更适用于高维度聚类。在对比二者所消耗的时间时,发现 KSNN-Chameleon 算法和 Chameleon 算法所消耗的时间几乎没有很大的差别。但在二维以上环境中,KSNN-Chameleon 算法所消耗的时间还是略小于传统算法的,这是由于在稀疏化阶段,KSNN-Chameleon 算法为了更好地适应高维聚类,在传统算法只使用 K 近邻图的基础上,又对数据进行了近邻加权的操作,所以增加了时间的消耗。同时,该算法为了抵消这一部分操作带来的时间影响,在 K 近邻法中又增加了 KD 树的概念,减少了部分内存和时间的多余损耗,甚至在高维情况下还有一定的提升。

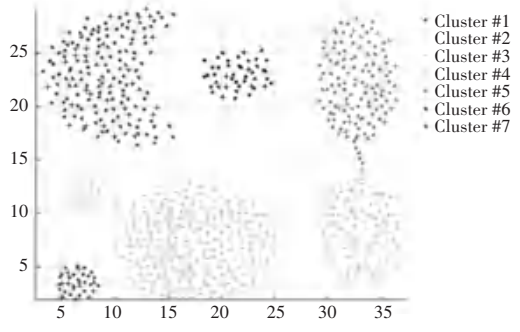

图 4　KSNN-Chameleon 算法对 Aggregation 数据集聚类结果

Fig. 4　The clustering results of Aggregation dataset using the KSNN-Chameleon algorithm

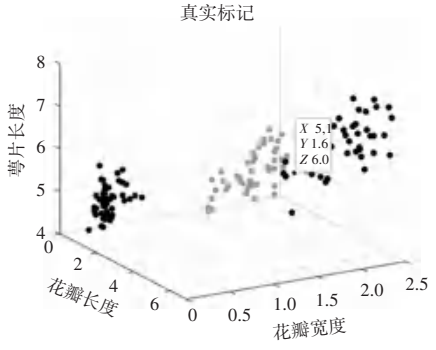

图 5　KSNN-Chameleon 算法对 Iris 数据集聚类结果

Fig. 5　The clustering results of Iris dataset using the KSNN-Chameleon algorithm

表 2　StuData 信息

Tab. 2　StuData information

学生性格	月生活费	生活习惯	作息时间	学习情况
热爱社交(57 人)	1 000 以下(41 人)	没有恶习(72 人)	10 点以前(32 人)	排名靠前(41 人)
热爱科学(67 人)	1 000~1 499(56 人)	不抽烟喝酒(90 人)	10~11 点(57 人)	排名中上(73 人)
热爱分享(49 人)	1 500~1 999(88 人)	抽烟不喝酒(42 人)	11~12 点(60 人)	排名中等(87 人)
尽职尽责(50 人)	2 000~3 000(65 人)	抽烟且喝酒(85 人)	12~凌晨 1 点 (73 人)	排名中下(58 人)
内向腼腆(66 人)	3 000 以上(39 人)		凌晨 1 点以后(67 人)	排名靠后(30 人)

3.2　实验条件和数据来源

实验测试采用的硬件环境如下：CPU 为 i5-8300H，内存为 8 GB。运行环境为 Windows 10 操作系统。软件环境为 IntelliJ IDEA 2020.3.4 x64、Matlab2020b。语言为 Matlab、JAVA。

3.3　数据集稀疏化

以 Matlab 中的二维数据集 Aggregation 和学生调查的五维数据集 StuData 为例，将二维数据集 Aggregation 展示在二维平面中，StuData 作为五维数据集将其中 3 个特征向量的数据抽取出来展示在三维坐标系中。结合第 1 节中的稀疏化步骤，针对数据集 Aggregation 取 k 值为 10，然后通过传统 Chameleon 算法以相似度计算得到的近邻图如图 6(a)所示，通过改进型 Chameleon 算法得到的加权近邻图如图 6(b)所示，可以看出两者在同样的 k 值和二维环境中得到的近邻图相似。按照同样的步骤，在五维数据集 StuData 中取 k 值为 10 后，通过传统算法得到的近邻图如图 6(c)所示，而通过 KSNN-Chameleon 算法得到的加权近邻图如图 6(d)所示。

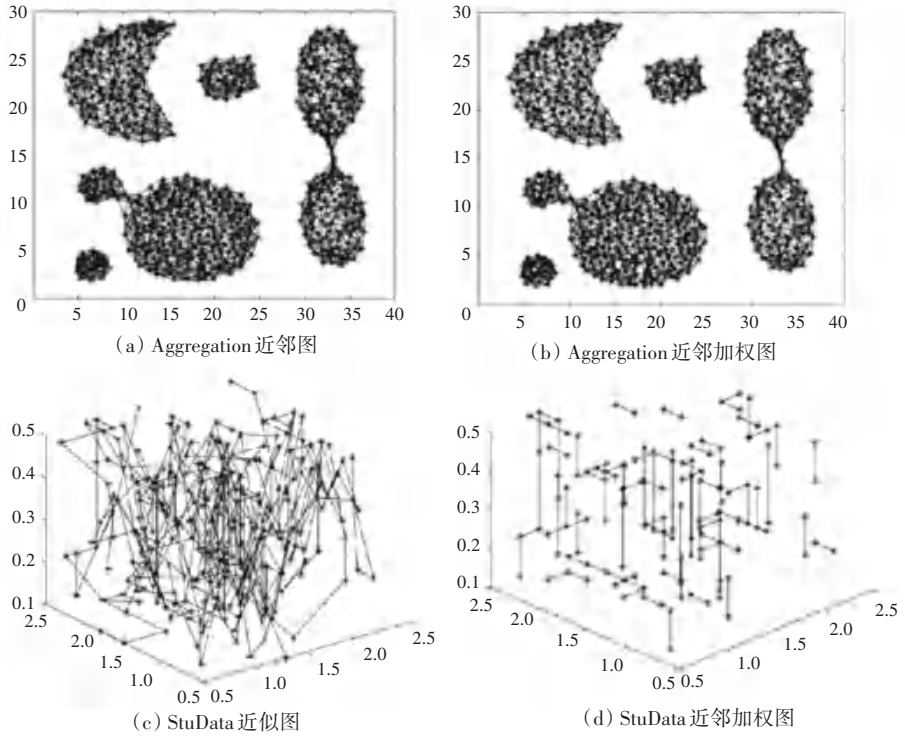

图 6　对不同数据集使用传统算法和改进算法稀疏化

Fig. 6　Using the traditional algorithm and the improved algorithm to sparsely process different data sets

由图 6 可以看出在高维度情况下，KSNN-Chameleon 算法的稀疏化结果更加精确，并且没有出现远点聚类情况。

3.4　通过 KSNN-Chameleon 宿舍分配

本文通过问卷调查的形式访问了 289 名在校学生的共 7 项意愿，从而整理得到了五维数据集 StuData。在通过对每一项指标详尽的分配权重后，对其采用多种不同形式的聚类算法进行聚类，模拟出宿舍分寝情况。采用聚类评估的方式，对这些聚类方法进行评估，验证这些算法在面对高维聚类时的性能优劣，对比结果见表 3。

表 3　采用不同算法对 StuData 进行聚类

Tab. 3　Clustering StuData with different algorithms

聚类算法	*AC*	*NMI*	*ARI*	时间/ms
KSNN-C	0.936	0.882	0.691	2.91
Chameleon	0.692	0.671	0.511	3.01
M-Chameleon	0.855	0.823	0.660	3.19
K-means	0.672	0.681	0.500	1.92
CURE	0.862	0.897	0.593	3.15
DPC	0.892	0.873	0.672	3.20

由表 3 可看出，将 KSNN-Chameleon 算法与目前几种主流聚类算法，包括：传统 Chameleon 算法、M-Chameleon 算法、K-means、CURE、DPC 算法，进行对比发现，针对五维数据集 StuData 和 KSNN-

Chameleon 算法的聚类精度是最高的。对 6 种算法的 *NMI* 值进行对比后,发现 KSNN-Chameleon 算法除了略低于 CURE 算法外, *NMI* 值都高于其它算法。对测试数据集重叠程度进行测量的 *ARI* 值进行比对后,发现 KSNN-Chameleon 算法依然拥有良好的表现,仅略低于 DPC 算法。在聚类时间上,由于 KSNN-Chameleon 算法在聚类过程中需要对稀疏图进行加权,所以运行的时间也相对延长了。对比传统 Chameleon 算法,KSNN-Chameleon 算法的聚类精度平均提升了 0.165, *NMI* 值平均提升了 0.205, *ARI* 值提升了 0.181,在面对高维 R 型聚类问题时明显更加稳定,证明在高维环境中 KSNN-Chameleon 算法能够进行更有效的聚类。

在对比多种算法后,本文采用 KSNN-Chameleon 算法对 StuData 的数据图和聚类结果图进行分析,图 7 为三维环境中展示数据集 StuData 的所有数据点,图 8 为使用传统 Chameleon 算法对数据进行聚类后的结果图。图 9 为通过 KSNN-Chameleon 算法对数据集进行聚类后的结果图。通过图 8 和图 9 的对比可以看出,通过 KSNN-Chameleon 算法得到的聚类图相较 Chameleon 聚类得到的结果聚类图更加均匀,每个簇中的数据也更加平均和接近。证明 KSNN-Chameleon 算法对传统算法的改良是有效的。

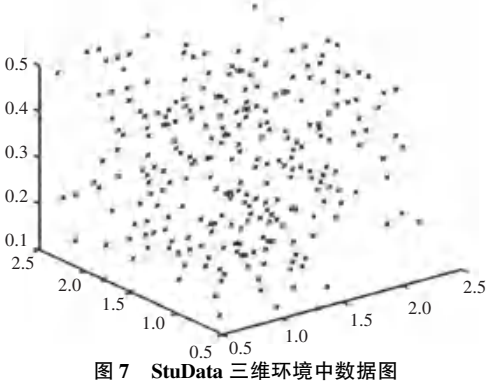

图 7 StuData 三维环境中数据图

Fig. 7 Data graph of StuData in 3D environment

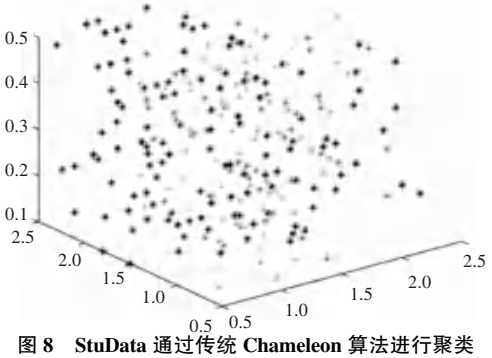

图 8 StuData 通过传统 Chameleon 算法进行聚类

Fig. 8 StuData clustering by traditional Chameleon algorithm

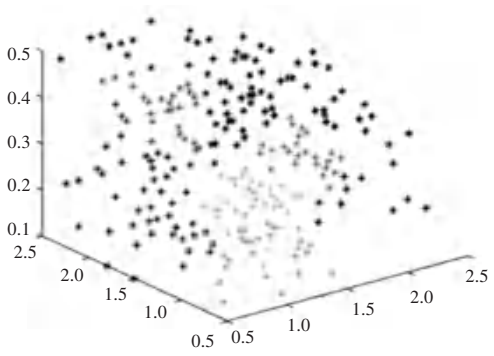

图 9 StuData 通过 KSNN-Chameleon 算法进行聚类

Fig. 9 StuData clustering by KSNN-Chameleon algorithm

研究针对 StuData 数据集中 289 名学生进行聚类后,将每一簇的学生按照每 4 人一组模拟为一个宿舍,绘制成宿舍分配表发放到这 289 名学生手中,并再次对学生进行调研回访。在 289 名学生中有65.05%的学生愿意按照此宿舍分配表的方式进行换寝,另外34.95%的学生认为维持原有的宿舍分配也不错。

4 结束语

本文针对传统的 Chameleon 算法进行改进,提出基于加权近邻法的 KSNN-Chameleon 算法,在针对 R 型高阶聚类问题时能够有着良好的聚类效果。算法首先在聚类的稀疏化阶段采用了加权近邻图的方法,有效避免了在 R 型聚类中使用距离为参照标准的问题。然后用洪水覆盖法(flood-fill)代替原有的 hMetis 算法,对加权近邻图的处理更加地细腻。最后利用共享近邻相似度和第一截断法,使得在高维环境中也能更好地将数据进行聚类,而不出现分散的问题。分析可知,KSNN-Chameleon 算法在聚类的准确率和精度方面对比传统算法均有所提高。KSNN-Chameleon 算法对比传统 Chameleon 算法的聚类精度提升了 20.88%,聚类时间上则提升了 2.73%,由此证明 KSNN-Chameleon 算法在面对高维 R 型聚类问题时更加稳定。

参考文献

[1] 新华网. 大数据“推荐算法”分宿舍可在所有高校推广[EB/OL]. [2018-08-27]. http://www.xinhuanet.com/comments/2018/08/27/c_1123332289.htm.

[2] 李梦园,肖理庆. 改进遗传算法大学生宿舍智能分配仿真研究[J]. 计算机仿真,2020,37(07):224-228.

[3] 罗恒,马圣尧,李明杰,等. 基于大数据技术的智能宿舍分配系统[J]. 信息与电脑(理论版),2020,32(08):51-52.

[4] KATHERINED, SUVRA P, SIDDIQUA J A. Stochastic EM algorithm for generalized exponential cure rate model and an empirical study[J]. Journal of Applied Statistics, 2021, 48(12): 2112-2135.

(下转第 36 页)