基于多叉树确定 K 值的动态 K-means 聚类算法

鲍黎明,黄 刚
(南京邮电大学 计算机学院,江苏 南京 210003)

摘 要:K-means 聚类算法是基于划分的经典聚类算法之一,因其简洁、高效得到了广泛的应用。K-means 算法具有容易实现、时间和空间复杂度较小的优点。但该算法的初始聚类数 K 通常不能通过有效的手段事先确定,其初始聚类中心往往是随机选取的,易收敛于局部最优解,造成聚类结果的不准确。基于多叉树确定 K 值的动态 K-means 聚类算法是对传统算法的改进,力求在迭代过程中动态分裂合并簇来确定最合理的聚类数,并且能在一定程度上解决聚类结果收敛于局部最优解的问题。文中还探索了相应的数据模型以支持所改进算法的研究,并从横向与纵向两方面与二分 K-means 算法作了对比实验。实验结果表明,改进后的 K-means 算法不依赖于全局数据集,更适用于分布式平台运算;算法相对效率随着数据集规模的增大,特别是在洪量数据集下具有明显的优势。

关键词:K-means;聚类;分裂;合并;多叉树
中图分类号:TP301.6 **文献标识码**:A **文章编号**:1673-629X(2017)06-0041-05
doi:10.3969/j.issn.1673-629X.2017.06.009

A Dynamic Clustering Algorithm of K-means Based on Multi-branches Tree for K-values

BAO Li-ming, HUANG Gang
(School of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

Abstract: K-means algorithm is the one of most classical clustering algorithms with repartition and has been widely used because it's really concise and efficient. What's more, it has advantages such as being easy to be implemented and low cost of complexity in running time and storing space. However, it's normally initial number called K-value which cannot be precisely predicted by effective method. The initial clustering center used to be chosen randomly, so that the result usually converges to local optimal solution, which makes the latest clustering results inaccurate. The dynamic clustering algorithm of K-means based on multi-branches tree to determine the K-value is an improved one. The improved algorithm has been designed to determine the most reasonable K-value by dynamically dividing and merging cluster during the iterative process and partly solved the problem that clustering result converges to local optimal solution. Furthermore, exploration for corresponding data structure model has also been conducted to the investigation of the algorithm mentioned. Horizontal and vertical comparison with the binary K-means algorithm has been achieved. The comparison and analysis results show that the improved K-means algorithm is independent of improved global data sets, which makes it more suitable for distributed computing platform and that relative efficiency has been increased with increase of the size of the data set, especially in magnanimity data set. Therefore the improved K-means algorithm has promoted the clustering performance and can lead to a more stable clustering result.

Key words: K-means; clustering; dividing; merging; multi-branches tree

0 引 言

大部分 K-means 聚类算法需要事先确定初始聚类数 k[1],k 值的选取在很大程度上会影响算法的性能。一般规律认为最佳的聚类数应该在 2 与 $\sqrt{N}$ 之间,其中 N 为数据集中所有数据元素的个数。许多学者都致力于确定 K-means 算法最优聚类数的研究,并提出了多种解决方法[2]。张忠平等提出了一种基于二分均值聚类的 k 值决定方法[3-5]。算法思想为:

(1)设定簇内相似度 λ 和簇间相似度 γ 两个阈值。

收稿日期:2016-05-13 **修回日期**:2016-09-14 **网络出版时间**:2017-03-13
基金项目:国家自然科学基金资助项目(61171053);南京邮电大学基金(SG1107)
作者简介:鲍黎明(1990-),男,研究方向为云计算与大数据应用;黄 刚,教授,研究方向为计算机软件理论及应用。

(2)在所选数据集上运行二分 K 均值聚类算法,得到两个类 C_1 和 C_2,计算 C_1 和 C_2 的簇内相似度 λ',若 $\lambda' > \lambda$,则继续运行二分 K 均值聚类算法,不断迭代以上过程,直到得到所有的簇内相似度都小于 λ。

(3)计算所有簇的簇间相似度,将簇间相似度小于 γ 的簇合并。

(4)得到的聚类个数即为 k 值。

该算法通过分裂和合并两个过程,得到较为准确的聚类结果,思想较为简单。二分 K 均值聚类算法是改进 K -means 算法思想的来源。

基于多叉树确定 k 值的动态聚类算法是一种不依赖于初始聚类数、在聚类过程中通过分裂与合并动态地确定 k 值的算法。基于多叉树确定 k 值的动态聚类算法不同于二分 K 均值聚类算法的最明显表现是,在迭代开始时可以在 2 与 N 之间随机指定初始聚类数,每次迭代可以分裂合并多个类簇。

这样聚类中心数即 k 值在每次迭代后是动态变化的,随着聚类中心趋于稳定,最终确定的 K 值即为最终的聚类数目。实验结果表明,基于多叉树确定 k 值的动态聚类算法确定的聚类数目与实际聚类数目相同,且算法的聚类准确性得到了相应提高。

1 基本思想

传统 K -means 算法[6]的基本步骤为:

(1)选择 k 个点作为初始质心。

(2)重复以下过程:将每个点指派到最近的质心,形成 k 个簇;重新计算每个簇的质心。

(3)直到质心不发生变化,算法结束。

基于多叉树确定 k 值的动态 K -means 聚类算法是对传统 K -means 算法的改进,但也离不开核心的迭代过程。该算法对于初始聚类数和初始质心的选取要求不高,但是合理地选取初始聚类数和初始质心可以降低算法的运行时间,提高效率。初始质心的选取可以采取基于密度的方法[7]来进一步提高聚类效率,然而,为凸显基于多叉树确定 k 值的动态聚类算法的有效性,利用随机均匀选择初始质心的方法。改进算法的迭代过程是区别于传统聚类算法和二分 K 均值聚类算法的核心所在。

图 1 为算法思想的直观阐述。

2 算法的简单性描述

基本定义:

定义 1:簇质心的计算公式为:

$$W_i = \left(\frac{1}{n_i}\sum i_{i1}, \frac{1}{n_i}\sum i_{i2}, \cdots, \frac{1}{n_i}\sum i_{ip}\right) \quad (1)$$

其中,n_i 为簇 C_i 内数据点的个数;$I_i = (i_{i1}, i_{i2}, \cdots, i_{ip})$ 为簇 C_i 内维度为 p 的数据对象。

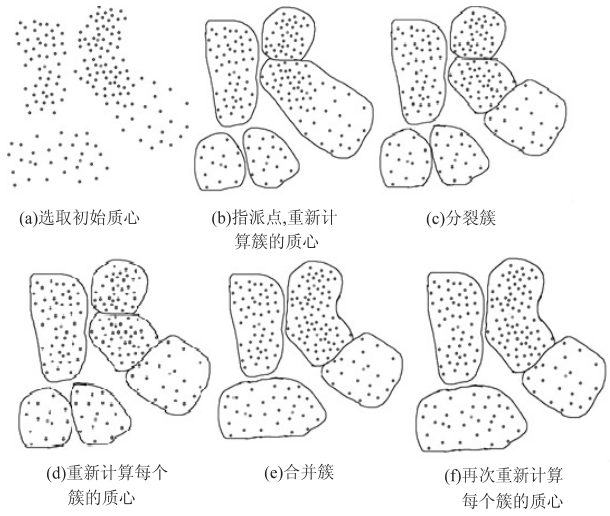

图 1 算法思想示意

定义 2:两个数据对象间的欧氏距离[8]为:

$$d(I_i, I_j) = \sqrt{|i_{i1} - i_{j1}|^2 + |i_{i2} - i_{j2}|^2 + \cdots + |i_{ip} - i_{jp}|^2} \quad (2)$$

其中,$I_i = (i_{i1}, i_{i2}, \cdots, i_{ip})$ 和 $I_j = (i_{j1}, i_{j2}, \cdots, i_{jp})$ 是两个 p 维的数据对象。

定义 3:簇中所有的数据点到簇质心的距离的平均值定义为簇内相似度[9],计算公式为:

$$\text{inner} = \frac{1}{n}\sum_{I \in C_i}\sqrt{|i_1 - w_{i1}|^2 + |i_2 - w_{i2}|^2 + \cdots + |i_p - w_{ip}|^2} \quad (3)$$

其中,n_i 为簇 C_i 内数据点的个数;$I = (i_1, i_2, \cdots, i_p)$ 为簇 C_i 内的数据元素;$W_i = (w_{i1}, w_{i2}, \cdots, w_{ip})$ 为簇 C_i 的质心。Inner 值越小,说明簇间相似性越大;反之,则越小。

定义 4:簇质心到其他簇质心的最小距离定义为簇间相似度[9],计算公式为:

$$\text{ext} = \min \| W_i - W_j \| = \min\sqrt{|w_{i1} - w_{j1}|^2 + |w_{i2} - w_{j2}|^2 + \cdots + |w_{ip} - w_{jp}|^2} \quad (4)$$

其中,$W_i = (w_{i1}, w_{i2}, \cdots, w_{ip})$ 与 $W_j = (w_{j1}, w_{j2}, \cdots, w_{jp})$ 分别为簇 C_i 与 C_j 的质心。ext 的值越大说明簇间相似度越小;反之,则越大。

基本步骤为:

(1)确定初始聚类数 k 并随机选取初始质心(见图 1(a))。

(2)将每个点指派到最近的质心,指派的标准为欧氏距离,重新计算每个簇的质心(见图 1(b))。

(3)分裂簇(见图 1(c))。

(4)重新计算每个簇的质心(见图 1(d))。

(5)合并簇(见图 1(e))。

(6)再次重新计算每个簇的质心(见图 1(f))。

从算法每一次的迭代步骤中会发现除了第六步重新计算质心外,中间还有两次需要计算质心,这是因为簇的分裂和合并所依赖的簇内相似度和簇间相似度都需要明确每个簇的质心,关于分裂与合并簇的标准,下面将会详细讨论。还会发现迭代过程是先分裂簇后合并簇,这是因为,分裂后的簇也许会符合合并簇指标,先分裂后合并会适当减少迭代次数,提高算法效率。还有一个问题是如何避免已分裂的簇在合并阶段又合并在一起的问题,这个问题会让迭代无限执行下去而得不到最终的聚类结果。因此,分裂后所得到的两个簇要做标识,来避免在合并阶段,将分裂的两个簇合并。

分裂标准:算法在分裂阶段如何正确选择该分裂的簇进行分裂是一个难题,因此算法需要一个簇内相似度阈值来判断哪些簇该分裂。

inner 呈正态分布[10]:$f(x)=\frac{1}{\sigma\sqrt{2\pi}}e^{\frac{(x-\mu)^2}{2\sigma^2}}$,$\mu$,$\sigma$ 的概率估计分别为:

$$\mu=\overline{\text{inner}}=\frac{1}{k}\sum_{i=1}^{k}\text{inner}_i \tag{5}$$

$$\sigma=\frac{1}{k}\sum_{i=1}^{k}(\text{inner}_i-\overline{\text{inner}})^2 \tag{6}$$

取拐点 b(见图 2)作为分裂阈值,当簇 inner $>\mu+\frac{1}{\sigma}$ 时,说明该簇的簇内相似度较小,应当进行分裂。

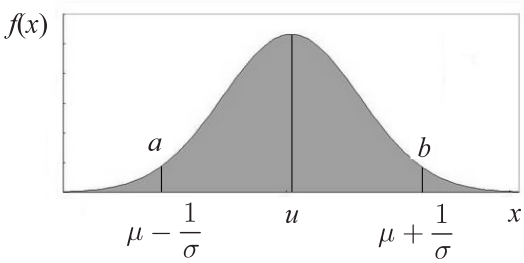

图 2　正态分布图

分裂算法过程:

输入:簇 inner;

输出:分裂的两个簇。

Begin

(1)While inner $>\mu+\frac{1}{\sigma}$ do

(2) $k=k+1$

(3)选择 2 个点作为初始质心。

repeat

(4)将每个点指派到最近的质心,形成 K 个簇。

(5)重新计算每个簇的质心

until 质心不发生变化

End

分裂算法具体过程其实就是将需分裂的簇视为一个原始样本点,在此簇内运用原始 K-means 算法进行聚类,唯一不同的是聚类数的取值为 2。当然分裂后的两个簇并不能保证它们是纯簇(只包含一类样本点的簇),但是基于多叉树确定 k 值的动态聚类算法会在迭代过程中不断使簇的元素数据趋于单一。

因为分裂簇后会造成聚类数增加 1,因此需要在原来 k 的基础上加 1。

合并标准如下:

两个簇要合并需依次满足三个标准才可进行:所谓依次是指满足了标准一,才可以去看是否满足标准二,进而标准三。

标准一:要合并的两个簇不可以是大簇分裂后形成的两个簇,下文将讨论如何识别两个簇是否为分裂后的小簇。

标准二:满足 ext 最大的两个簇,满足标准二说明簇间相似度较大,且相邻。

标准三:取拐点 a(见图 2)作为簇内相似度的阈值,即簇 inner $<\mu-\frac{1}{\sigma}$ 时,满足标准三,说明该簇簇内相似性过小。

因为合并簇后会造成聚类数减 1,因此需要在原来 k 的基础上减 1。

3　基于多叉树的数据结构模型

因为基于多叉树确定 k 值的动态聚类算法涉及到簇的分裂合并,特殊的簇比如分裂后的簇需要标识,来避免在后面合并的部分又让其合并。此外分裂合并会造成聚类数的变化,也就是说 k 值是不断浮动变化的,因此还需保证 k 值的实时有效性。综合多方面考虑,数据节点的结构如下:

Static int k;

Class Node{

Int flag = i; $i\in(1,2,\cdots,k)$

Int $*$ $k=k$;

Int inner,ext;

Int P_flag = p; p 为父节点的标识,根节点 p 为 0

Node $*$ L_child = L;

Node $*$ R_child = R; 当节点本身为叶节点时,即未分裂时 L 和 R 为空

}

静态全局变量 k 为聚类数,这样可以保证 k 的唯一性与实时性,因为在分裂时 k 值要加一,合并要减一,并且下一步迭代依赖于 k 的真实值。Flag 为此质心节点的唯一标识。P_flag 为父节点的标识,簇分裂后形成的两个小簇的 P_flag 的值相同,用来避免合并

步骤又重新合成。Inner 与 ext 表示本簇的簇内与簇间相似度。

L_child 与 R_child 为指向本簇分裂后所形成的两个小簇的指针,这里根节点有所不同,因为初始聚类数 k 值并不是 2,所以根节点应有初始 k 个子节点。

4 算法具体分析

第一步需要定义一个目标函数或准则函数作为迭代停止的标准[11-13]。因为基于多叉树确定 k 值的动态聚类算法是基于 K -means 传统算法,因此它也具备随着不断的迭代质心趋于稳定,即规则函数 $E = \sum_{i=1}^{k} \sum_{I_j \in c_i} |I_j - W_i|^2$ 收敛的特征。但是,基于多叉树确定 k 值的动态 K -means 聚类算法规则函数 E 收敛的同时,分裂合并的活动会相应减少,图 2 所示的 $(\mu - \frac{1}{\sigma}) < \text{inner} < (\mu + \frac{1}{\sigma})$ 域中数据所占比重会逐渐增加到峰值,所以改进算法的目标函数有如下定义: $w_{ab} = \int_{u-\frac{1}{\sigma}}^{u+\frac{1}{\sigma}} f(x)\,\mathrm{d}x$, $w = \int_{-\infty}^{+\infty} f(x)\,\mathrm{d}x$。目标函数[14]为 $E_w = \frac{w_{ab}}{w}$,即当 E_w 收敛时,算法迭代结束。

变量定义: k 为该算法在数据集上输出的聚类数量; n 为数据集对象元素个数。在初始化时,从数据集 $\{I_1, I_2, \cdots, I_n\}$ 随机找出 k 个 $\{W_1, W_2, \cdots, W_k\}$, $W_i = I_j$, $i \in \{1,2,\cdots,k\}$, $j \in \{1,2,\cdots,n\}$ 作为簇的初始均值或中心,对剩余的每个对象,根据其与各个簇均值(见式(1))的距离(见式(2)),将它指派到最相似的簇,计算每个簇的新均值。执行分裂、确定质心、合并、确定质心、再指派这个过程,不断反复,直到准则函数 E_w 收敛,或者分裂与合并停止。

基于多叉树确定 k 值的动态 K -means 聚类算法的描述如下:

输入:包含 n 个对象的数据集及簇的数目 k;

输出:簇的集合。

Begin

初始化 k 个簇中心 $\{W_1, W_2, \cdots, W_k\}$,其中 $W_i = I_j$, $i \in \{1,2,\cdots,k\}$,使每一个聚类 C_j 与簇中心 W_j 相对应

Repeat

For 每一个输入向量 I_j, $j \in \{1,2,\cdots,n\}$ do ①

①将 I_j 分配给最近的簇中心 W_i^* 所属的 $C_i{}^*$,即 $\| I_j - W_i^* \| = \min \| I_j - W_i \|$, $i \in \{1,2,\cdots,L\}$, L 为每次分裂合并迭代后聚类数的新值, L 的初始值 $L = K$

②For 每一个聚类 C_i, $i \in \{1,2,\cdots,L\}$,do ③

③将簇中心更新为当前的 C_j 中所有样本的中心点,即 $W_i = \sum_{I_j \in C_i} I_j / |C_i|$,并且计算基于新中心点的 inner 和 ext

④当簇 inner > $\mu + \frac{1}{\sigma}$ 时分裂簇

⑤For 每一个分裂后的类 C_m,其中 m 指上一步被分裂的簇产生的小簇 do⑥

⑥将簇中心更新为当前 C_m 中所有样本的中心点,即 $W_m = \sum_{I_j \in C_m} I_j / |C_m|$,并且计算基于新中心点的 inner 和 ext

⑦当簇满足合并标准时合并簇

⑧For 每一个合并后的类 C_b,其中 b 指上一步被合并的簇产生的合簇 do ⑨

⑨将簇中心更新为当前 C_b 中所有样本的中心点,即 $W_b = \sum_{I_j \in C_b} I_j / |C_b|$

⑩计算目标函数 E_w

Until E_w 不再明显改变或者聚类的成员不再变化

End

5 实 验

验证在不同的数据集及不同的初始聚类中心选取数目和选取方式因素下,实验算法都能高效地得到准确的聚类结果。

影响实验结果的因素有数据集大小、初始聚类中心选取数目、初始聚类中心选取方式。为了验证算法的有效性,实验选取了数个规模不同的先验数据集,在这些数据集上分别进行了不同初始聚类中心选取数目和方式的基于多叉树确定 k 值的动态 K -means 聚类算法(算法 A)的实施。为了验证算法的高效性,与基于二分均值聚类的 k 值决定方法(算法 B)进行了对比,通过比较聚类结果以及聚类时间来验证实验的高效性。

实验需要横向比较与纵向比较两大步骤。横向比较是指在同一数据集初始聚类中心选取数目与选取方式的差异所导致的算法的优效比;纵向比较是指在不同规模数据集上的算法优效比。

实验算法的核心是分裂合并两个步骤,因此这两个步骤的迭代次数是算法效率的最直接体现,所以实验选取分裂合并次数和聚类时间作为验证算法优劣的参数。

为了使初始聚类中心选取数目(K_0)的选取具有代表性, K_0 的选取有较少、适量、较多三个层次。因为所选取数据集为先验数据集, K_0 的适量层次即为准确的聚类数 K;较少层次首先需要包括 1,其余需要从 1-

K 之间均匀选取三个数值;较多层次需包括最大聚类簇元素数 N,以及 $K-N$ 之间均匀选取三个数值。

初始聚类中心选取方式包括局部分布、均匀分布两种。

横向比较实验结果见表 1。

表 1　横向比较结果

初始 K 值	初始聚类中心分布方式	分裂次数	合并次数	最终聚类数	聚类时间/ms
1	局部分布	4	1	5	33
	均匀分布	4	0	5	30
2	局部分布	6	3	5	30
	均匀分布	3	0	5	13
3	局部分布	8	6	5	37
	均匀分布	4	2	5	14
4	局部分布	4	3	5	31
	均匀分布	4	3	5	8
5	局部分布	5	5	5	33
	均匀分布	0	0	5	5
6	局部分布	6	7	5	39
	均匀分布	0	1	5	5
7	局部分布	4	6	5	36
	均匀分布	0	1	5	7
8	局部分布	7	10	5	35
	均匀分布	0	3	5	5
9	局部分布	8	12	5	35
	均匀分布	0	4	5	8

分析以上结果,可以得到如下结论:

(1)当 K_0 选取过少时,聚类迭代时分裂次数明显增加,聚类所需时间明显增加。

(2)当 K_0 选取过多时,聚类迭代时合并次数有所增加,聚类时间也会相应增加,但聚类花费时间明显小于当初始 K 值过少时的聚类所花费时间。

(3)当 K_0 选取合适,但初始聚类中心局部分布时,迭代过程中合并次数明显增加,聚类时间明显增加。

(4)当 K_0 选取合适,且均匀分布时,分裂合并次数在正常范围内,聚类时间最短。

以上四种情况,实验聚类算法运行正常,聚类结果正确。

纵向比较实验结果:实验需控制初始聚类中心选取数目及选取方式两个变量。实验初始聚类中心选取数目 K_0 等于最终聚类数 K,选取方式为均匀分布。实验结果如图 3 所示。

图中,横坐标为数据集规模(以最终聚类数为参数),纵坐标为随着数据集规模的增长时分裂次数、合并次数、聚类时间的量化。

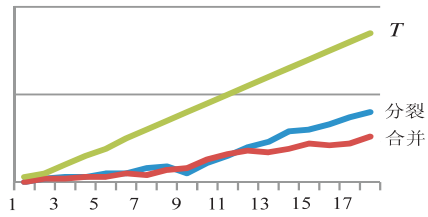

图 3　实验结果图

由图 3 可知,随着聚类规模的递增,分裂次数与合并次数增长趋势平缓,呈现非指数型增长,聚类时间呈线性增长。证明算法在不同规模数据集的时间复杂度在可接受范围内。

最后实验使用外部数据集 Wine data 和 Iris data 对算法 A 与算法 B 进行了对比,结果见表 2。

表 2　对比实验

算法	Wine data		Iris data	
	T	K	T	K
算法 A	31.53	8	40.15	8
算法 B	38.15	8	48.33	8

实验结果表明,基于多叉树确定 k 值的动态 K-means 聚类算法在 Wine data 和 Iris data 数据集所用时间分别是基于二分均值聚类的 k 值决定算法的 82.6% 和 83.1%,说明改进算法相较于基于二分均值聚类的 k 值决定方法的效率有所提高。

以上实验说明,基于多叉树确定 k 值的动态 K-means 聚类算法对于各种不同的初始 k 值选取、不同的聚类中心选取方式,都能在可控时间范围内,得到正确的聚类结果,并且具有较高的效率。

6　结束语

K-means 算法是广泛应用的聚类算法之一,通过研究分析传统 K-means 算法的局限性,用一种新思路对 K-means 算法进行适当改进,使改进算法不再严重依赖初始聚类数的选取。实验结果表明,改进算法在聚类准确性和效率上都有适当提升。当然,算法还有进一步改进的可能,比如可在分布式文件系统中实现并行化,如 Hadoop 平台可使算法效率进一步提高,这是下一步研究的方向。

参考文献:

[1] Han J W, Wen S P. Data mining: concepts and techniques [M]. San Francisco: Morgan Kaufmann Publishers, 2000.

[2] 杨善林,李永森,胡笑旋,等. K-means 算法中的 k 值优化问题研究[J]. 系统工程理论与实践, 2006, 26(2): 97-101.

[3] Wang Aijie, Chai Xuguang. Easy and efficient algorithm to

(下转第 50 页)

[2] 刘 勃,张在峰,马义德,等. 基于分形理论的图像压缩编码技术[J]. 信息与电子工程,2004,2(4):246-251.
[3] 尹显东,唐 丹,邓 君,等. 图像小波变换的分形编码技术[J]. 信息与电子工程,2003,1(3):23-27.
[4] Li J,Kuo C C J. Image compression with a hybrid wavelet-fractal coder[J]. IEEE Transactions on Image Processing,1999,8(6):868-874.
[5] 陈明夫. 基于区域检测的小波分形图像压缩方法[D]. 哈尔滨:哈尔滨理工大学,2013.
[6] Davis G M. A wavelet-based analysis of fractal image compression[J]. IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society,1998,7(2):141-154.
[7] 杜广环. 基于小波分析的图像压缩算法应用[D]. 大连:大连理工大学,2008.
[8] 向 辉. 基于小波理论的图像压缩算法研究[D]. 上海:华东师范大学,2005.
[9] 郝俊瑞,许红军. 基于分类的小波-分形混合编码[J]. 无线电工程,2001(z1):85-88.
[10] 张爱华,盛 飞,杨 培,等. 基于相似比的快速分形编码算法[J]. 计算机技术与发展,2012,22(11):176-178.
[11] Zhang Y,Zhai G. Wavelet-based fractal image compression[C]//Third international symposium on multispectral image processing and pattern recognition. [s. l.]:International Society for Optics and Photonics,2003:396-399.
[12] 黄 晋. 混合小波-分形图像压缩方法的研究与实现[D]. 贵阳:贵州大学,2007.
[13] Barnsley M,Vince A. Fractal tilings from iterated function systems[J]. Discrete & Computational Geometry,2014,51(3):729-752.
[14] Davoine F,Robert G,Chassery J M. How to improve pixel-based fractal image coding with adaptive partitions[M]// Fractals in engineering. London:Springer,2001:292-306.
[15] Barnsley M F,Hurd L P. Fractal image compression[M]. Wellesley:AK Peters,1992.
[16] Wohlberg B,Jager G. A review of the fractal image coding literature[J]. IEEE Transactions on Image Processing,1999,8(12):1716-1729.
[17] Jacobs E W,Fisher Y,Boss R D. Image compression:a study of the iterated transform method[J]. Signal Processing,1992,29(3):251-263.
[18] Fisher Y. Fractal image compress:theory and application[M]. New York:Spring-Verlag,1995:49-51.
[19] 庄振静,何传江,申小娜. 基于规范块半范数的快速分形编码算法[J]. 计算机工程与应用,2010,46(2):170-173.
[20] 徐 庆,刘 弘,吴晓燕. 基于 2-范数匹配的分形图像编码改进算法[J]. 计算机工程,2010,36(4):205-206.
[21] 李高平. 分形几何及其在图像压缩编码中的应用研究[D]. 重庆:重庆大学,2005.
[22] Shi Yipen,Gu Wei,Zhang Liming. Some new methods to fractal image compression[J]. Communication in Nonliner Science & Numerical Simulation,1997,13(2):80-85.
[23] Polvere M,Nappi M. Speedup in fractal image coding:comparison of methods[J]. IEEE Transactions on Image Processing,2000,9(6):1002-1009.
[24] 黄小虎,胡 清,黄 杰. 基于 MATLAB 的分形仿真研究[J]. 电脑知识与技术,2007(5):847-849.

(上接第页 45)

determine number of clusters[J]. Computer Engineering and Applications,2009,45(15):166-168.
[4] Lai J Z C,Huang T J. Fast global k-means clustering using cluster membership and inequality[J]. Pattern Recognition,2010,43(5):1954-1963.
[5] Zhang Zhongping,Steinbach M,Karypis G,et al. A comparison of document clustering techniques[R]. USA:University of Minnesota,2000.
[6] 冯 超. *K*-means 聚类算法的研究[D]. 大连:大连理工大学,2007.
[7] 谢娟英,郭文娟,谢维信,等. 基于样本空间分布密度的初始聚类中心优化 k-均值算法[J]. 计算机应用研究,2012,29(3):888-892.
[8] 宋宇辰,张玉英,孟海东. 一种基于加权欧氏距离聚类方法的研究[J]. 计算机工程与应用,2007,43(4):179-180.
[9] 元昌安. 数据挖掘原理与 SPSS Clementine 应用宝典[M]. 北京:电子工业出版社,2009.
[10] 陆声链,林士敏. 基于距离的孤立点检测研究[J]. 计算机工程与应用,2004,40(33):73-75.
[11] 苏锦旗,薛惠锋,詹海亮. 基于划分的 K-均值初始聚类中心优化算法[J]. 微电子学与计算机,2009,26(1):8-11.
[12] 步媛媛,关忠仁. 基于 K-means 聚类算法的研究[J]. 西南民族大学学报:自然科学版,2009,35(1):198-200.
[13] 马 帅,王腾蛟,唐世渭,等. 一种基于参考点和密度的快速聚类算法[J]. 软件学报,2003,14(6):1089-1095.
[14] 孙吉贵,刘 杰,赵连宇. 聚类算法研究[J]. 软件学报,2008,19(1):48-61.