文章编号:1671-9352(2026)01-0065-11 DOI:10.6040/j.issn.1671-9352.0.2024.353

结合蝙蝠算法和紧密度改进的三支 *K*-means 算法

孙清[1,2],叶军[1,2*],曾广财[1,2],宋苏洋[1,2],汪一心[3]
(1.江西水利电力大学信息工程学院, 江西 南昌 330000; 2.智慧水利江西省重点实验室, 江西 南昌 330000; 3.江西开放大学,江西 南昌 330000)

摘要:本文结合蝙蝠算法和紧密度改进三支 *K*-means 算法,利用黄金分割系数和种群平均位置优化蝙蝠算法,根据优化后的蝙蝠算法搜索初始聚类中心,提高三支 *K*-means 算法的稳定性。依据紧密度判断核心域和边界域的阈值,减少边界域样本数量,提高三支 *K*-means 算法的准确性。对比实验采用 9 个数据集与 6 种聚类算法,实验结果表明本文算法提升聚类性能,验证本文算法有效性和实用性。

关键词:*K*-means 聚类;蝙蝠算法;紧密度;*K*-means 算法;三支决策
中图分类号:TP391 **文献标志码**:A
引用格式:孙清,叶军,曾广财,等. 结合蝙蝠算法和紧密度改进的三支 *K*-means 算法[J]. 山东大学学报(理学版),2026,61(1):65-75.

Three-way *K*-means algorithm combining the bat algorithm and the improved compactness

SUN Qing[1,2], YE Jun[1,2*], ZENG Guangcai[1,2], SONG Suyang[1,2], WANG Yixin[3]
(1. Colege of lnformation Engimeering, Jiangxi University of Water Resources and Electric Power, Nanchang 330000, Jiangxi, China; 2. Jiangxi Province Key Laboratory of Smart Water Conservancy, Nanchang 330000, Jiangxi, China; 3. Jiangxi Open University, Nanchang 330000, Jiangxi, China)

Abstract: The three way *K*-means algorithm is improved by integrating the bat algorithm with closeness degree optimization. The bat algorithm is optimized by employing the golden section coefficient and population average position . The optimized bat algorithm searches for initial cluster centers which improving the stability of the three way *K*-means algorithm. Additionally, the threshold for core and boundary regions is determined based on closeness degree, which reduces the number of boundary samples and enhances the accuracy of the three way *K*-means algorithm. Comparative experiments is conducted on nine datasets against six clustering algorithms. It is shown that the proposed method improves clustering performance and is confirming its effectiveness and practical utility.

Key words: *K*-means clustering; bat algorithm; compactness; *K*-means algorithm; three way decision

0 引言

聚类分析[1]是无监督学习算法中重要的研究内容之一,聚类分析包括划分聚类[2]、层次聚类[3]、密度聚类[4]、网格聚类[5]等。传统聚类方法处理的样本数据只能二选一,即属于某个类或不属于某个类。传统的二支聚类方法无法准确判定信息不充分数据的类别归属,生活中的某些信息不充分的数据或方案同样无法准确作出决策,为此,文献[6-7]中提出三支决策理论,这种理论的核心思想是将决策集一分为三,接受决策、

收稿日期:2024-10-17; **网络出版时间**:2025-10-10
基金项目:江西省教育厅科技基金资助项目(GJJ211920); 国家自然科学基金资助项目(62566041)
第一作者:孙清(1999—),男,硕士研究生,研究方向为粗糙集理论、聚类方法、群体智能优化算法等. E-mail:2310680106@qq.com
*通信作者:叶军(1968—),男,教授,硕士生导师,硕士,研究方向为知识发现与数据挖掘、粗糙集与粒计算理论.
E-mail:2003992646@nit.edu.cn

延迟决策和拒绝决策。分别用核心域、边界域和琐碎域表示分类对象,文献[8-9]将三支理论引入聚类算法,提出三支聚类方法。文献[10]中在三支 *K*-means 算法中引入聚类有效性指数,提出自动三支决策的 *K*-means 算法。文献[11]中设定阈值 α,对簇类数 K 进行动态调整,提出三支动态阈值的 *K*-means 算法。文献[12]利用样本稳定性将数据集划分为稳定数据集和不稳定数据集,得到三支聚类的核心域和边界域,提出基于邻域样本稳定性的三支聚类算法。文献[13]分析了样本稳定性的合理性,基于信息熵提出样本稳定性的度量方法。文献[14]在文献[13]的基础上将样本相似性引入三支 *K*-means 算法中,提出新的聚类有效性指标,比较聚类有效性指标,最大聚类数目作为最佳聚类数目,提出基于稳定性和相似性的三支聚类算法。文献[15]将朴素贝叶斯共现概率引入三支聚类算法中,利用共现概率的相似关系得到核心域和边界域,提高聚类精度。

上述三支聚类算法过于依赖初始中心点的选择,容易陷入局部最优,为此,研究者引入群智能优化算法寻找聚类中心,例如,文献[16]将人工蜂群算法引入粗糙 *K*-means 聚类算法,利用迭代次数的自适应阈值加快了收敛速度,改进后的算法提高了聚类精度并减少迭代次数。文献[17]将变异的萤火虫算法引入粗糙 *K*-means算法,改善原算法中初始中心点敏感性和稳定性不高等问题。文献[18]将变异的萤火虫算法引入三支聚类算法中,利用 q 近邻的概念得到边界域样本的数量,使边界域中满足阈值条件的样本尽可能地划分到核心域中。文献[19]利用人工蜂群算法中蜂群之间不同角色的合作和角色互换的特点,寻找初始聚类中心,提高聚类性能。文献[20]将蚁群聚类算法与三支决策融合,得到三支决策的蚁群聚类算法,提升聚类效果和时间效率。文献[21]中调整粒子群算法中的速度参数,使粒子在迭代过程中能较快地找出全局最优解,即最优的聚类中心,基于粒子群的三支聚类算法具有较高的全局最优性能和聚类的准确性。

基于群智能优化算法改进的三支 *K*-means 聚类算法,如基于粒子群、萤火虫和蚁群等改进的聚类算法,有效降低了算法对初始聚类中心的依赖性,并减少了算法迭代次数。但是,这些算法对核心域和边界域采用阈值划分,导致核心域和边界域划分模糊,部分样本的类别归属确定困难,出现核心域和边界域错分的情况,并且这些算法在计算复杂度、快速搜寻最优聚类中心等方面仍须改进。

蝙蝠算法[22-24]是模仿蝙蝠在狭窄空间中导航的搜索算法,该算法参数少、实现简单、收敛速度快,适应于求解连续型优化问题。本文将蝙蝠算法和紧密度引入三支 *K*-means 聚类算法,提出基于黄金分割系数和种群平均位置引导蝙蝠搜索最优位置改进的蝙蝠算法,依据样本紧密度判断核心域和边界域,利用样本紧密度改进适应度函数的权重。

1 预备知识

1.1 三支聚类理论

设样本集为 $U=\{x_1,x_2,\cdots,x_n\}$,簇类中心 $M=\{m_1,m_2,\cdots,m_K\}$,依据簇类中心得到的聚类结果 $C_K=\{C_{m_1},C_{m_2},\cdots,C_{m_K}\}$, d 表示样本的属性个数,其中 n 表示样本个数,K 表示簇类个数。

三支决策的核心域 C_{core}、边界域 C_{fringe} 和琐碎域 C_{trivial} 是类的 3 个部分。核心域表示正决策域,样本确定属于该类;琐碎域表示负决策域,样本确定不属于该类;边界域表示延迟决策域,样本无法确定属于哪类。得到的每个类为 $C=\{(C_{\text{core}}^1,C_{\text{fringe}}^1),(C_{\text{core}}^2,C_{\text{fringe}}^2),\cdots,(C_{\text{core}}^k,C_{\text{fringe}}^k),\cdots,(C_{\text{core}}^K,C_{\text{fringe}}^K)\}$,其中 $k\in[1,K]$, C_{core}^k 表示第 k 类的核心域,C_{fringe}^k 表示第 k 类的边界域,核心域、边界域和琐碎域的关系为 $C_{\text{core}}\subset U$, $C_{\text{fringe}}\subset U$, $C_{\text{trivial}}=U-(C_{\text{core}}\cup C_{\text{fringe}})$。

1.2 蝙蝠算法

蝙蝠算法是模拟蝙蝠捕食的群智能优化算法。蝙蝠使用回声定位寻找猎物,蝙蝠发出响亮的脉冲并通过周围物体反弹的回声确定物体的方向和位置。蝙蝠算法具体过程如下。

步骤 1 初始化参数。蝙蝠种群规模 N,最大迭代次数 T,蝙蝠位置 $\boldsymbol{X}_i^t$,表示第 i 个蝙蝠在时间 t 的位置,$i\in[1,N]$, $t\in[1,T]$,速度 $\boldsymbol{v}_i^t$,声波频率 f_i,声波响度 A_i^t 和声波频度 r_i^t。

步骤 2 按照更新规则更新每个蝙蝠的位置,更新规则为

$$f_i=f_{\min}+(f_{\max}-f_{\min})\beta, \tag{1}$$

$$\boldsymbol{v}_i^t=\boldsymbol{v}_i^{t-1}+(\boldsymbol{X}_i^t-\boldsymbol{X}_*)f_i, \tag{2}$$

$$\boldsymbol{X}_i^{t+1}=\boldsymbol{X}_i^t+\boldsymbol{v}_i^t, \tag{3}$$

式中:$f_{\min}$、$f_{\max}$表示蝙蝠种群中最小声波频率和最大声波频率,β 是范围内服从均匀分布的随机向量,$\beta\in[0,1]$,X_* 为当前全局最优位置。在蝙蝠算法搜索过程中,蝙蝠由式(1)随机生成声波频率,得到声波频率后,由式(2)得到蝙蝠的速度 $\boldsymbol{v}_i^t$, $\boldsymbol{v}_i^t$ 是一个向量,决定了蝙蝠更新位置的方向与距离,最后由式(3)完成位置的更新。

步骤 3　生成随机数 R_{rand1}。

if $R_{\text{rand1}}>r_i^t$

则在当前最优解附近进行局部搜索,并得到新解

$$\boldsymbol{X}_{\text{new}}=\boldsymbol{X}_*+\boldsymbol{\xi}\bar{A}^t, \tag{4}$$

式中:$\boldsymbol{\xi}$ 是随机数向量,$\boldsymbol{\xi}$ 中随机数取值范围为[-1,1], $\bar{A}^t$ 为所有蝙蝠在相同时间段的平均响度。

步骤 4　生成随机数 R_{rand2}。

if $R_{\text{rand2}}>r_i^t$ and $\boldsymbol{X}_{\text{new}}$优于 $\boldsymbol{X}_i^t$

更新蝙蝠位置,即 $\boldsymbol{X}_i^{t+1}=\boldsymbol{X}_{\text{new}}$,并更新蝙蝠声波响度和声波频度,新位置的声波响度和声波频度分别为

$$A_i^{t+1}=\alpha A_i^t, \tag{5}$$

$$r_i^{t+1}=r_i^0[1-\exp(-\gamma t)], \tag{6}$$

式中:α 是声波响度衰减系数,$\alpha\in[0,1]$,γ 是声波频度增强系数,$\gamma>0$,r_i^0 表示蝙蝠 i 的初始声波频度。

步骤 5　重复步骤 2、3、4,获得最优解或达到最大迭代次数为止。

2　结合蝙蝠算法和样本紧密度改进的三支 *K*-means 聚类算法

2.1　蝙蝠算法的优化

标准的蝙蝠算法具有算法收敛速度快、实现简单、参数少等优点,但也存在蝙蝠位置易陷入局部最优、蝙蝠种群个体多样性差等问题。文献[25]在蝙蝠算法的全局随机飞行搜索中引入个体最优因素,增加路径搜索的发散性。文献[26]引入变步长搜索策略和二元素优化方法搜索局部,使得算法具有较高寻优能力和较强的实用价值。但上述改进算法还存在不足:由于算法使用的是全局最优解,容易使算法在当前全局最优位置附近徘徊,无法指向最优解;在算法前期通过全局最优解更新蝙蝠位置,虽然加快了收敛速度,但也导致搜索范围较为局限;算法收敛速度过快,且种群搜索不具有多样性,使得算法容易陷入局部最优而无法跳出。本文从以下 2 个方面改进蝙蝠算法。

2.1.1　引入黄金分割系数

在数学和优化领域,黄金分割法[27]常用于单峰函数的一维搜索,其核心是通过不断缩小搜索区间找到极值点。

从式(1)—(3)可以看出,主要通过速度和频率调整蝙蝠的位置更新,本文将黄金分割系数 τ 引入蝙蝠算法,利用黄金分割率调整位置更新的步长,在全局搜索阶段有较大的步长,而在局部搜索阶段步长逐渐减小,利用黄金分割的比例控制步长的变化,有效地覆盖搜索空间,增大了搜索空间。新个体的更新位置和更新速度分别为

$$\boldsymbol{X}_i^{t+1}=\boldsymbol{X}_i^t|\sin R_1|+\boldsymbol{v}_i^t, \tag{7}$$

$$\boldsymbol{v}_i^t=R_2\sin R_1|a_1\boldsymbol{p}_i^t-a_2\boldsymbol{X}_i^t|, \tag{8}$$

式中:R_1、R_2 是随机数,$R_1\in[0,2\pi]$, $R_2\in[0,\pi]$,a_1 和 a_2 为调节系数,$a_1=-\pi+2\pi(1-\tau)$、$a_2=-\pi+2\pi\tau$,τ 为黄金分割系数,$\boldsymbol{p}_i^t$ 为第 i 个蝙蝠到第 t 时刻为止的最优位置。当种群进行迭代时,经典蝙蝠算法按照式(2)、(3)更新蝙蝠位置,式(2)中 f_i 与 v_i^t 均是指向全局最优,搜索范围较小,容易陷入局部最优。式(8)中去掉了式(2)中的 f_i 频率参数,改为由 R_2 和 R_1 共同控制蝙蝠速度。a_1、a_2 用于调控个体搜索空间,根据需要设置不同的值,以提高搜索效率。且式(7)与式(3)相比,引入随机数 R_1 进行位置更新,更新后的位置不再

是一个固定位置,是在一个范围内随机选择,扩大了蝙蝠的搜索范围,防止蝙蝠位置陷入局部最优。经典蝙蝠算法搜索范围和引入黄金分割系数改进后的算法的搜索范围如图 1、2 所示。

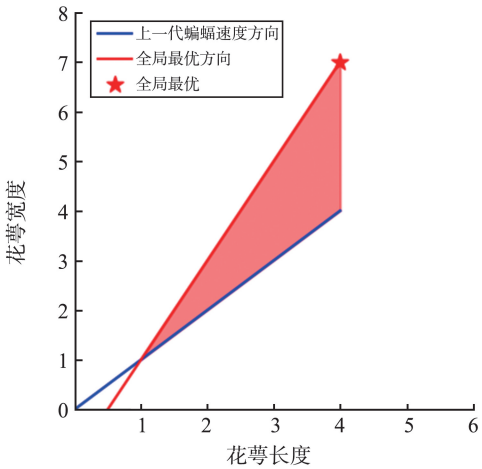

图 1 经典蝙蝠算法搜索范围

Fig.1 Classical bat algorithm search range

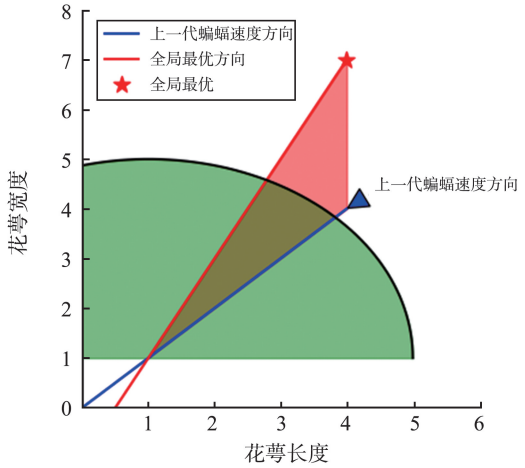

图 2 改进后蝙蝠算法搜索范围

Fig.2 Classical bat algorithm search range

图 1、2 为一个蝙蝠的寻优搜索范围,蝙蝠寻优的数据集为降维的 Iris 数据集。如图 1 所示,红色区域为经典蝙蝠算法搜索区域,该区域为上一代蝙蝠速度方向和蝙蝠与最优解之间方向的夹角区域,虽然在该区域内蝙蝠个体适应度快速收敛,由于搜索区域较小导致蝙蝠位置陷入局部最优。如图 2 所示,使用黄金分割系数优化后的蝙蝠算法,虽然增加了蝙蝠路径的搜索广度,但蝙蝠整体搜索方向大致不变,搜索效率损失不大。

2.1.2 种群平均位置引导搜索

由于标准的蝙蝠算法搜索策略是按照全局最优进行搜索,在变异机制方面灵活度较小,为此,本文引入种群平均位置对个体进行搜索引导,增加种群的多样性,种群平均位置定义如下。

定义 1 t 时刻蝙蝠的平均位置 $\boldsymbol{m}^t$ 为

$$\boldsymbol{m}^t = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{X}_i^t, \tag{9}$$

平均位置更新的速度为

$$\boldsymbol{s}_i^t = R_2 \sin R_1 \left| a_1 \boldsymbol{m}^t - a_2 \boldsymbol{X}_i^t \right|, \tag{10}$$

由种群平均位置和最优个体位置引导蝙蝠个体搜索如图 3 所示。

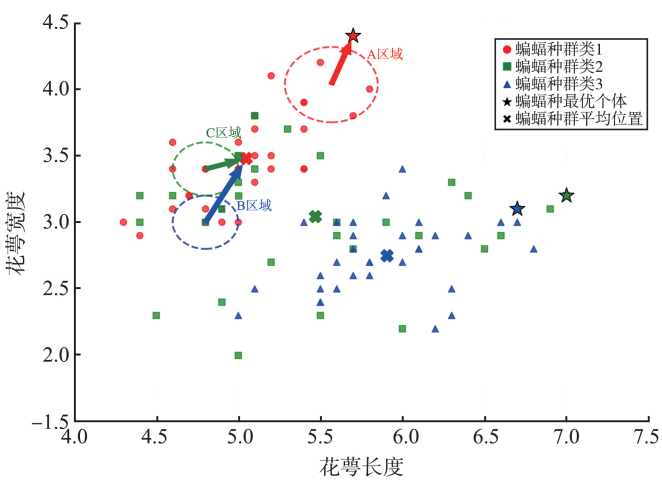

图 3 种群平均位置引导与最优个体引导

Fig.3 Population average position guidance and optimal individual guidance

图 3 是部分蝙蝠在二维空间中取随机值后迭代过程中的情况。蝙蝠迭代寻优的数据集为降维的 Iris 数据集,二维空间区域为 $x \in [4.0, 8.0]$, $y \in [2.0, 4.5]$。可以看出,当大部分种群个体例如 A 区域中的蝙蝠都

围绕在最优蝙蝠位置附近时,若所有蝙蝠都以最优蝙蝠引导其他蝙蝠迭代,容易陷入局部最优。为了防止陷入局部最优,本文将蝙蝠种群按照适应度值分为前、后部分,使用不同的搜索策略,对适应度值高的前半部分群体采用式(8)搜索,向着图中最优蝙蝠搜索,加快收敛速度,如图中 A 区域的蝙蝠中靠近最优蝙蝠;而对于适应度值低的后半部分群体,而适应度较低的后半部分蝙蝠种群通过式(10)迭代,向着图中种群平均位置方向搜索,防止所有蝙蝠都被最优蝙蝠吸引导致陷入局部最优,有效增加种群的多样性,例如图中 B、C 区域中远离最优蝙蝠的蝙蝠。

2.2　引入样本紧密度划分对象

大多数三支聚类中的核心域和边界域的界限都较为模糊,文献[10]中区分核心域和边界域按照类中对象到中心的距离的最大差值区分核心域和边界域,如果类中对象分布较为均匀,各个对象到中心的距离差别不大,导致核心域和边界域的区分不明确。文献[16]直接划分或按照迭代次数变化确定阈值,但人为给定阈值没有考虑类和类之间的差异以及每个类中不同对象的分布情况,可能会降低对象划分的准确率。本文借鉴文献[12-13]中样本稳定性相关知识,定义样本紧密度,通过样本紧密度区分样本数据的核心域和边界域。

定义 2　设样本集合 $U=\{x_1,x_2,\cdots,x_n\}$,近邻表示离对象 x 最近的 q 个对象的集合,记作 $N_q(x)$,2 个对象的 q 近邻集合相交的个数与 q 的比值定义为共现概率,即

$$p_{\mathrm{gh}}=\frac{|\{N_q(x_g)\cap N_q(x_h)\}|}{q},\tag{11}$$

式中:n 表示样本个数,$g\in[1,n]$,$h\in[1,n]$,共现概率表示 2 个样本中邻域交集的个数与邻域个数的比值。p_{gh}越大表示样本 x_g 和样本 x_h 关系更紧密,反之,关系越疏远。

定义 3　设第 k 个簇类的样本集合 $C_{m_k}=\{x_1,x_2,\cdots,x_z\}$,则样本 x_g 的紧密度为

$$J_g=\frac{1}{z}\sum_{h=1}^{z}p_{\mathrm{gh}},\tag{12}$$

式中,z 为第 k 个簇中对象个数。紧密度衡量集合中 x_g 和其他样本的紧密关系。集合的核心域中的样本与其他样本的关系更加紧密,就像人际关系中的群体核心成员与大多数成员保持紧密关系,因此,计算每个类别中对象的紧密度,能确定对象的归属。若 x_g 紧密度高,说明 x_g 与其他样本关系紧密,将 x_g 划入核心域类别中;反之,则说明 x_g 在该类别中与其他样本联系较为疏远,则划入边界域中,即核心域 C^k_{core}与边界域 C^k_{fringe}分别为

$$C^k_{\mathrm{core}}=\{x_g\,|\,J_g\geqslant\varepsilon,1\leqslant g\leqslant z\},\tag{13}$$

$$C^k_{\mathrm{fringe}}=\{x_g\,|\,J_g<\varepsilon,1\leqslant g\leqslant z\},\tag{14}$$

式中:ε 为划分核心域边界域的阈值,在算法前期,样本的归属关系不明确,所以较大的 ε 可以使样本尽可能划入到各个类边界域中。随着迭代次数的增加,样本的归属关系也逐渐明确,较小的 ε 可以将样本尽可能划入到各个类的下近似中,所以,ε 随着算法的迭代动态变化的,文献[17]中引入 levy 曲线,得到

$$\varepsilon=\sqrt{\frac{c}{2\pi}(t-\mu)^3\mathrm{e}^{\frac{c}{(\mu-t)}}},\tag{15}$$

式中:$c=\frac{1}{2}$,$\mu=\frac{1}{2}$。

2.3　构造适应度函数

与其它群智能优化算法一样,蝙蝠算法中的种群同样由适应度函数值引导蝙蝠进行搜索,适应度函数决定整个种群的搜索方向,本文定义类内聚集度函数和类间距离函数构造适应度函数。

定义 4　类内聚集度为

$$I=\sum_{k=1}^{K}\Big(\sum_{u=1}^{|C^k_{\mathrm{core}}|}w_u d(x_u,m_k)+\sum_{f=1}^{|C^k_{\mathrm{fringe}}|}w_f d(x_f,m_k)\Big),\tag{16}$$

式中:w_u、w_f 分别是核心域样本 x_u 和边界域样本 x_f 的权重,$w_u=f(J_u)$,$w_f=f(J_f)$,J_u 为样本 x_u 的紧密度,J_f 为样本 x_f 的紧密度,f 为权值函数,$f=(J_g-b)/(c-b)$,$b\leqslant J_g\leqslant c$,$f$ 将紧密度进行标准化映射到(b,c)中,b、c 为 f 的边界值。类内聚集度反应了同一类别中样本在中心周围的聚集程度。I 越小,代表中心周围聚集程度

越高,该中心的聚类效果越好。但文献[10,18]中的类内聚集度只考虑了样本距离因素,没有考虑样本分布状况。本文定义 2 所定义的紧密度反映了样本分布状况。

式(16)中,紧密度越大则数据样本对于簇类的重要性越大,反之重要性越小,因此,式(16)得到的类内聚集度 I 不仅体现了类中所有样本距离因素,而且反映类中不同样本分布的差异性。在数据样本个数相同的情况下,类内聚集度越小,则类中数据样本围绕中心周围分布越紧凑,样本相似度越高,聚类效果越好;反之,数据样本分布稀疏,相似度低,聚类效果差。

定义 5 类间离散度定义为

$$D=\sum_{j=1}^{K}\omega\sum_{l=j+1}^{K}d(m_j,m_l), \tag{17}$$

式中:ω 是权重系数,$\omega=\dfrac{1}{K}$,m_j,m_l 为聚类中心,$j,l\in[1,K]$。

式(17)中采用了文献[17]中权重系数平衡类内聚集度和类间离散度之间的距离,有效避免离群或孤立点的影响。类间离散度反应了类和类之间的关系,类间离散度越大说明类与类之间交集越少,差异性越大,聚类效果也越好。

定义 6 目标函数

$$f_X=\left(\frac{D}{I}\right)^{\eta}, \tag{18}$$

式中:η 是系数,η 能够防止类簇中样本过多而导致各个样本之间的总距离的和过大。D 越大,说明类簇与类簇之间的差异越大;I 越小,类间样本的相似性越高,因此,目标函数越大,聚类效果越好。

2.4 算法设计

算法 1 基于蝙蝠算法优化的三支 K-means 算法。

输入 U,N,T,K 。

输出 $C_K=\{(C_{\text{core}}^1,C_{\text{fringe}}^1),(C_{\text{core}}^2,C_{\text{fringe}}^2),\cdots,(C_{\text{core}}^k,C_{\text{fringe}}^k),\cdots,(C_{\text{core}}^K,C_{\text{fringe}}^K)\}$。

(1) 初始化蝙蝠种群及 K 个聚类中心;

(2) 计算所有样本间的 p_{gh};

(3) while $t<T_{\max}$

(4) 得到第 1 次聚类 $C_K=\{(C_{\text{core}}^1,C_{\text{fringe}}^1),(C_{\text{core}}^2,C_{\text{fringe}}^2),\cdots,(C_{\text{core}}^k,C_{\text{fringe}}^k),\cdots,(C_{\text{core}}^K,C_{\text{fringe}}^K)\}$;

(5) 计算所有对象的紧密度;

(6) 计算每个样本对应的权值;

(7) 根据式(13)、(14)、(15)得到阈值,对数据进行划分得到所有的$(C_{\text{core}},C_{\text{fringe}})$;

(8) 初始化蝙蝠的参数;

(9) for $i=1$ to K

(10) 由式(18)计算各个蝙蝠适应值;

(11) 将蝙蝠分为适应度高、低 2 类;

(12) for $j=1$ to N

(13) 蝙蝠按照式(7)—(10)更新;

(14) 更新每个蝙蝠的位置和参数;

(15) end for

(16) 更新适应度最高蝙蝠;

(17) end for

(18) 更新所有样本的紧密度;

(19) 更新每个样本对应权值;

(20) 计算所有样本的适应值,并更新中心;

(21) 按得到的新聚类中心重复步骤(7),得到新聚类 $C'_K=\{(C_{\text{core}}^1,C_{\text{fringe}}^1),(C_{\text{core}}^2,C_{\text{fringe}}^2),\cdots,(C_{\text{core}}^k,C_{\text{fringe}}^k),\cdots,(C_{\text{core}}^K,C_{\text{fringe}}^K)\}$;

(22)　继续迭代,直到达到最大迭代次数或中心不再变化;

(23)　end while

(24)　输出聚类结果。

2.5　算法时间复杂度分析

传统的 *K*-means 算法时间复杂度为 $O(kTn)$,本文算法的时间复杂度主要体现在数据的共现概率。步骤(1)、(2)、(3)中计算共现概率的时间复杂度为 $O(n^2)$, 步骤(6)、(7)计算紧密度的时间复杂度为 $O(2n)$,步骤(11)的时间复杂度为 $O(N)$,步骤 (21)、(22)更新紧密度的时间复杂度为 $O(2n)$,步骤(22)的时间复杂度为 $O(n)$,算法总时间复杂度为 $O((n^2+2n)+(N+3n)kT)d)$,时间复杂度量级为 $O(n^2)$。

文献[12-13]时间复杂度为 $2n^2+n\log n+|S|2\log|S|$,S 为稳定样本集,$|S|$为稳定样本集的样本个数。本文算法时间复杂度与文献[12-13]算法相比,没有明显差距,但随着样本数 n 的增大,本文算法与文献[13]的差距会增大。文献[10]的时间复杂度为 $O(n\log n+knq)$,由于本文算法需要计算紧密度,相比文献[10],本文算法的时间复杂度较高,但聚类平均准确率上大于文献[10]算法、分类适确性指数上小于文献[10]算法。

3　实验结果分析

为验证本文改进算法的有效性和可行性,本文选取加州大学欧文分校机器学习(University of CaliforniaIrvine Irvine Machine Learning Repository)数据库中 9 个常用数据集进行测试,数据集信息如表 1 所示。9 个数据集既有中、小样本和大容量样本数据集,也包括了中、小维和高维数据集,例如,Bank 数据集是高维且大容量,Waveform、Student 和 Rice 数据集是中维大容量,Wines 数据集是中维中、小容量,Abalone 数据集是低维大容量。测试指标为平均准确率、分类适确性指数、平均轮廓系数和各算法平均运行时间,将本文算法与以下 6 种算法进行对比:基于 *K*-means 的自动三支决策聚类算法[10]、基于邻域样本稳定性的三支聚类算法[12]、基于样本稳定性的聚类算法[13]、融合黄金正弦的蝙蝠算法[24]、融合变异萤火虫算法的三支聚类算法[18]、基于粒子群的三支聚类算法[21],上述 6 种算法分别称为算法 1、算法 2、算法 3、算法 4、算法 5、算法 6。算法 1 为经典的三支 *K*-means 算法,算法 2、3 是改进的三支聚类方法,算法 4、5、6 均融合了群体智能算法的三支聚类算法,其中,算法 4 与本文算法都是利用蝙蝠算法优化三支聚类算法。算法 1 因随机选取初始中心而影响聚类结果的稳定性和准确性,导致迭代次数多等问题,本文算法及算法 2—6 改进算法主要工作都是对算法 1 进行改进,因此,本文算法与上述 6 种算法具有可比性。

表 1　9 个数据集信息

Table 1　Nine data sets informations

序号	数据集	样本数	维数	类别数
1	Wine	178	13	3
2	Contraceptive	1 473	9	3
3	Abalone	4 177	7	3
4	Bank	4 521	72	2
5	Waveform	5 000	40	3
6	Ionsphere	351	32	2
7	Rice	3 810	32	2
8	Student	4 424	36	3
9	Winequalityred	1 599	11	6

试验运行环境:Win11 操作系统,编程软件为 Pycharm 2023.1.2,处理器为 Inter i7 12650,显卡为 RTX 4060。

当 $N=60$, $T=100$, $\tau=\sqrt{5}-\frac{1}{2}$, $\eta=\frac{1}{2}$时,本文算法的聚类平均准确率较高。重复实验 30 次,9 个数据集平均准确率见图 4 所示。由图 4 可知,$b=0.2$, $c=0.8$ 时本文算法获得的平均准确率最高。

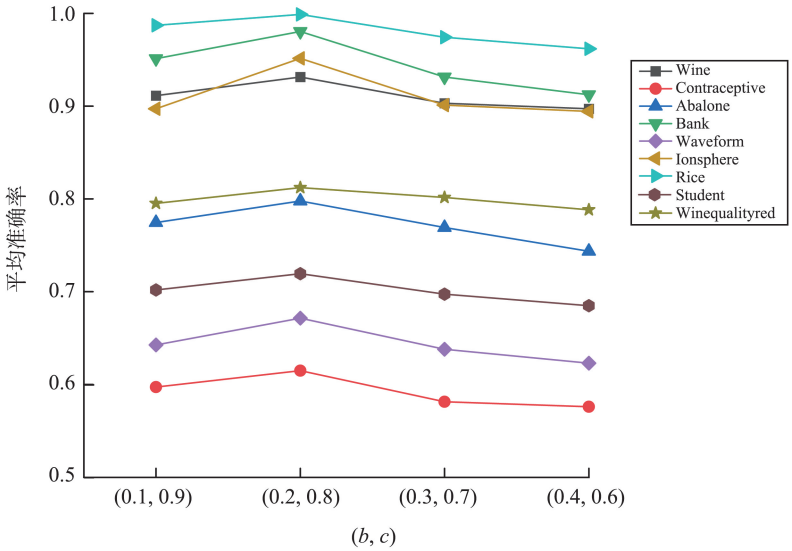

图 4 b、c 不同时 9 个数据集的平均准确率

Fig.4 The average accuracy rates of the nine datasets with b and c different

3.1 算法平均准确率对比

每个数据集运行 30 次,7 种算法的平均准确率结果如表 2 所示。

表 2 7 种算法的平均准确率

Table 2 Average accuracy of seven algorithms

序号	算法 1	算法 2	算法 3	算法 4	算法 5	算法 6	本文算法
1	0.897 2	0.913 3	0.927 2	0.928 2	0.925 7	0.855 3	**0.931 4**
2	**0.580 2**	0.411 2	0.430 2	0.448 4	0.435 3	0.413 9	0.404 8
3	0.567 4	0.619 1	0.622 8	0.656 8	0.678 1	0.662 8	**0.697 7**
4	0.659 9	0.676 2	**0.691 0**	0.684 8	0.657 9	0.647 3	0.670 2
5	0.642 2	0.627 4	0.638 2	0.660 8	0.653 1	0.641 3	**0.671 4**
6	0.751 6	0.760 6	0.773 8	0.791 3	0.783 9	0.775 9	**0.801 5**
7	0.921 3	0.925 6	0.921 3	0.947 3	0.937 7	0.933 5	**0.954 9**
8	0.585 9	0.619 1	0.597 7	**0.662 6**	0.650 7	0.649 3	0.659 3
9	0.329 1	0.331 2	0.338 6	0.326 5	0.330 1	0.322 4	**0.332 2**

由表 2 可知,采用 Contraceptive、Bank 和 Student 数据集时,本文算法的平均准确率比其他算法小,采用 Wine、Abalone、Waveform、Ionsphere、Rice 和 Winequalityred 数据集时,本文算法平均准确率结果大于其余 6 种算法,说明本文算法的稳定性高和聚类效果好。Contraceptive、Bank 数据集的数据分布存在类别不均衡情况,Student 数据集 3 个类别较均衡,但数据分布存在 0~100 分等极端情况,与 Wine、Abalone、Waveform、Ionsphere、Rice 和 Winequalityred 数据集相比本文算法平均准确率较大,说明虽然本文算法在某些类别不平衡和有噪声点数据集的平均准确率较小。

3.2 分类适确性指数对比

本文算法与其余 6 种算法平均分类适确性指数如图 5 所示。分类适确性指数[28]是一种判断聚类好坏的指标。平均分类适确性指数越小,代表聚类效果越好。由图 5 可知,采用 Abalone、Rice 数据集时,本文算法的分类适确性指数略小于算法 4。原因是 Abalone 数据集为混合数据类型,蝙蝠算法的优化对连续型数据集效果显著,但对离散型数据的效果一般,计算分类适确性指数时样本间距离不够显著,而 Rice 数据集中的某些特征有较高的特征相关性,导致冗余信息过多。而采用本文算法搜索时,由于搜索范围扩大,因此使得搜索效率下降并且进一步影响了聚类结果。但除了 Abalone、Rice 数据集以外,本文算法的平均分类适确性指数比其他算法小,说明本文算法在大多数情况能够提高聚类精度。

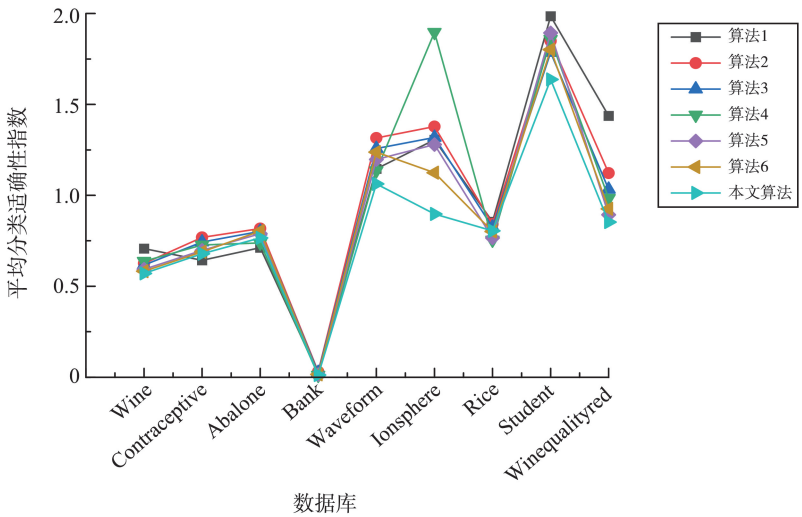

图 5　采用不同数据集时 7 种算法的平均分类适确性指数

Fig. 5　Average Davies-Bouldin index of seven algorithms with different data sets

3.3　平均轮廓系数对比

轮廓系数[29]是评价聚类好坏的指标,轮廓系数在 [-1,1] 之间,轮廓系数越大,说明该类中的样本属于该类簇的可能性越大,效果越好。

如图 6 所示,采用 Ionsphere 数据集时,本文算法相比算法 1 的轮廓系数值略小。Ionsphere 数据集的特点是维度较高且类别不平衡。蝙蝠算法在处理这类数据集时,有时会将本该属于核心域部分数据划分到簇类边界域,导致轮廓系数值较低。采用其他 8 个数据集时,本文算法均比其他算法的轮廓系数大。

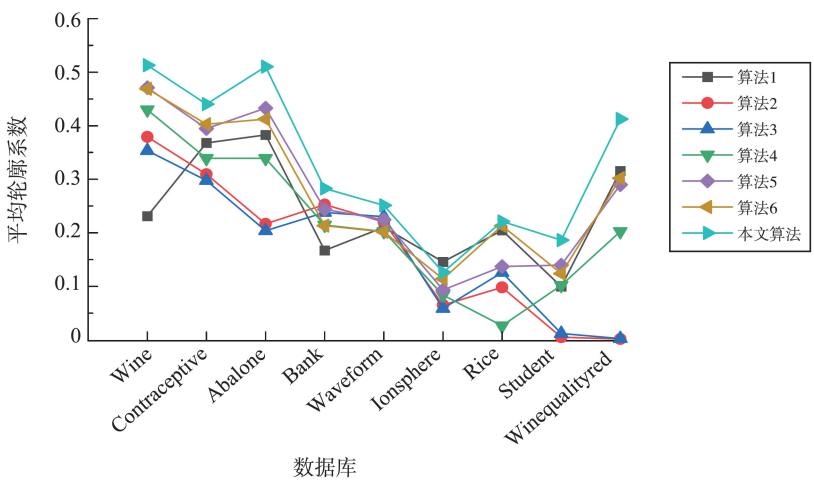

图 6　采用不同数据集时 7 种算法的平均轮廓系数

Fig.6　Average silhouette coefficient of 7 algorithms with different data sets

3.4　平均运行时间对比

采用不同数据集时,7 种算法的平均运行时间如表 3 所示,采用 Ionsphere、Winequalityred、Wine、Bank、Rice、Contraceptive 数据集时,本文算法的平均运行时间比算法 1、3、4、5 多,由于本文算法每次迭代需要计算紧密度,增加了计算量。采用 Abalone、Waveform、Student 数据集时,本文算法的平均运行时间少于算法 2、3、5、6,由于算法 2、3 每次迭代需要计算共现概率,算法 5 计算 q 近邻概念确定边界域时间复杂度,与本文算法计算量相当,而算法 6 的时间复杂度随着规模的增大按指数方式增长,当数据样本较大时,本文算法与算法 6 相比,平均运行时间更少。下一步研究尝试使用并行方式计算来减少时间消耗。

表 3 7 种算法在各个数据集上的平均运行时间

Table 3 The average running time of 7 algorithms on each data set 单位:s

序号	算法 1	算法 2	算法 3	算法 4	算法 5	算法 6	本文算法
1	0.315 7	0.478 2	0.465 7	0.223 1	1.374 7	1.465 2	1.596 8
2	1.597 5	21.773 0	23.584 0	4.1807 0	20.741 0	25.314 0	23.713 0
3	10.539 0	87.449 0	90.653 0	36.016 0	89.398 0	115.647 0	86.022 0
4	10.911 0	67.068 0	75.283 0	5.832 7	70.983 0	105.368 0	78.593 0
5	27.514 0	183.540 0	190.550 0	4.438 2	172.390 0	243.970 0	180.670 0
6	0.477 8	1.683 1	1.977 8	1.377 6	1.859 3	1.921 1	1.895 8
7	7.183 3	605.480 0	611.190 0	4.977 3	642.314 0	789.479 0	613.700 0
8	34.536 0	18.455 0	19.596 0	13.293 0	16.987 0	18.654 0	17.778 0
9	2.578 3	24.912 0	25.674 0	5.177 4	27.598 0	30.598 0	26.771 0

4 结语

本文利用黄金分割系数和种群平均位置对蝙蝠算法优化,扩大了蝙蝠种群的搜索范围,增加了种群多样性。改进后的蝙蝠算法搜索初始聚类中心,能提高算法的稳定性,通过紧密度判断核心域和边界域的阈值,并且能界定核心域和边界域。在蝙蝠算法中利用适应度函数搜索。实验结果表明,利用蝙蝠算法改进的三支聚类方法提高了聚类效果。虽然在搜索聚类中心过程中减少了迭代次数,但每次迭代要计算紧密度,增加了计算量,算法运行消耗时间较多。如何减少计算量是本文下一步的研究工作。

参考文献:

[1] SAMBASIVAM S, THEODOSOPOULOS N. Advanced data clustering methods of mining web documents[J]. Issues in Informing Science and Information Technology, 2006, 5(3):563-579.

[2] 章永来,周耀鉴. 聚类算法综述[J]. 计算机应用,2019,39(7):1869-1882.
ZHANG Yonglai, ZHOU Yaojian. Review of clustering algorithms[J]. Journal of Computer Applications, 2019,39(7):1869-1882.

[3] 张雨如. 基于动态建模的层次聚类算法研究[D]. 徐州:中国矿业大学,2022.
ZHANG Yuru. Research on hierarchical clustering using dynamic modeling [D]. Xuzhou: China University of Mining and Technology, 2022.

[4] 陈叶旺,申莲莲,钟才明,等. 密度峰值聚类算法综述[J]. 计算机研究与发展,2020,57(2):378-394.
CHEN Yewang, SHEN Lianlian, ZHONG Caiming, et al. Survey on density peak clustering algorithm[J]. Journal of Computer Research and Development, 2020, 57(2):378-394.

[5] 马福民,宫婷,杨帆,等. 基于 Zipf 分布的网格密度峰值聚类算法[J]. 控制与决策,2024,39(2):577-587.
MA Fumin, GONG Ting, YANG Fan, et al. Grid density peak clustering algorithm based on zipf distribution[J]. Control and Decision, 2024, 39(2):577-587.

[6] YAO Yiyu, PAWAN Lingras, WANG Ruizhi, et al. Interval set cluster analysis: a reformulation[C] //Rough Sets, Fuzzy Sets, Data Mining and Granular Computing. Berlin: Springer, 2009:398-405.

[7] YAO Yiyu. The superiority of three way decisions in probabilistic rough set models[J]. Information Sciences, 2011, 181(6): 1080-1096.

[8] 于洪,王国胤,姚一豫. 决策粗糙集理论研究现状与展望[J]. 计算机学报,2015,38(8):1628-1639.
YU Hong, WANG Guoyin, YAO Yiyu. Current research and future perspectives on decision-theoretic rough sets[J]. Chinese Journal of Computers, 2015, 38(8):1628-1639.

[9] YU Hong, ZHANG Cong, WANG Guoyin. Overlapping clustering method using the three-way decision theory[J]. Knowledge Based Systems, 2016, 91:189-203.

[10] 于洪,毛传凯. 基于 *k*-means 的自动三支决策聚类方法[J]. 计算机应用,2016,36(8):2061-2065.
YU Hong, MAO Chuanka. Automatic three-way decision clustering algorithm based on *k*-means[J]. Journal of Computer Applications, 2016, 36(8):2061-2065.

[11] 解滨,董新玉,梁皓伟. 基于三支动态阈值 *K*-means 聚类的入侵检测算法[J]. 郑州大学学报(理学版),2020,52(2):64-70.
XIE Bin, DONG Xinyu, LIANG Haowei. An algorithm of intrusion detection based on three-way dynamic threshold *K*-means clustering[J]. Journal of Zhengzhou University(Natural Science Edition), 2020, 52(2):64-70.

[12] 李洪梅,姜冬勤,王平心. 基于邻域样本稳定性的三支聚类方法[J]. 山西大学学报(自然科学版),2020,43(4):874-879.
LI Hongmei, JIANG Dongqin, WANG Pingxin. Three-way clustering based on neighborhood sample's stability[J]. Journal of Shanxi University(Natural Science Edition), 2020, 43(4):874-879.
[13] 李飞江,钱宇华,王婕婷,等. 基于样本稳定性的聚类方法[J]. 中国科学(E 辑:信息科学),2020,50(8):1239-1254.
LI Feijiang, QIAN Yuhua, WANG Jieting, et al. Clustering method based on sample's stability[J]. Science in China(Series E: Information Sciences), 2020, 50(8):1239-1254.
[14] 李浩溥. 基于稳定性和相似性的三支聚类算法研究[D]. 哈尔滨:哈尔滨师范大学,2023.
LI Haobo. Three way *K*-means algorithm based on sample stability and similarity[D]. Harbin: Harbin Normal University, 2023.
[15] 花遇春,赵燕,马建敏. 基于共现概率的三支聚类模型[J]. 西北大学学报(自然科学版),2022,52(5):797-804.
HUA Yuchun, ZHAO Yan, MA Jianmin. Three-way clustering model based on co-occurrence probability[J]. Journal of Northwest University(Natural Science Edition), 2022, 52(5):797-804.
[16] 叶廷宇,叶军,王晖,等. 结合人工蜂群优化的粗糙 *K*-means 聚类算法[J]. 计算机科学与探索,2022,16(8):1923-1932.
YE Tingyu, YE Jun, WANG Hui, et al. Rough *K*-means clustering algorithm combined with artificial bee colony optimization [J]. Journal of Frontiers of Computer Science & Technology, 2022, 16(8):1923-1932.
[17] 李兆彬,叶军,周浩岩,等. 变异萤火虫优化的粗糙 *K*-均值聚类算法[J]. 山东大学学报(工学版),2023,53(4):74-82.
LI Zhaobin, YE Jun, ZHOU Haoyan, et al. A rough *K*-means clustering algorithm optimized by mutation firefly algorithm [J]. Journal of Shandong University(Engineering Science), 2023,53(4):74-82.
[18] 李兆彬,叶军,周浩岩,等. 融合变异萤火虫算法的三支聚类方法[J]. 系统仿真学报,2025,37(3):646-656.
LI Zhaobin, YE Jun, ZHOU Haoyan, et al. Three-way decision clustering algorithm fusion of mutant fireflies algorithm[J]. Journal of System Simulation, 2025, 37(3):646-656.
[19] 徐天杰,王平心,杨习贝. 基于人工蜂群的三支 *K*-means 聚类算法[J]. 计算机科学,2023,50(6):116-121.
XU Tianjie, WANG Pingxin, YANG Xibei. Three-way *K*-means clustering based on artificial bee colony[J]. Computer Science, 2023, 50(6):116-121.
[20] 王梦绚,万仁霞,苗夺谦,等. 基于三支决策的蚁群聚类算法[J]. 昆明理工大学学报(自然科学版),2024,49(1):83-97.
WANG Mengxun, WAN Renxia, MIAO Duoqian, et al. An ant colony clustering algorithm based on three way decision[J]. Journal of Kunming University of Science and Technology(Natural Science), 2024, 49(1): 83-97.
[21] 高艳龙,万仁霞,陈瑞典. 基于粒子群的三支聚类算法[J]. 福州大学学报(自然科学版),2022,50(3):301-307.
GAO Yanlong, WANG Renxia, CHEN Ruidian. A three-way clustering algorithm based on particle swarm optimization[J] Journal of Fuzhou University(Natural Science Edition), 2022, 50(3):301-307.
[22] YANG Xinshe. A new metaheuristic bat-inspired algorithm[J]. Computer Knowledge & Technology, 2010, 284:65-74.
[23] 许德刚,赵萍. 蝙蝠算法研究及应用综述[J]. 计算机工程与应用,2019,55(15):1-12.
XU Degang, ZHAO Ping. Literature survey on research and application of bat algorithm[J]. Computer Engineering and Applications, 2019, 55(15): 1-12.
[24] 倪昌浩,邹海. 基于改进蝙蝠算法的移动机器人路径规划方法研究[J]. 制造业自动化,2021,43(6):53-56,62.
NI Changhao, ZOU Hai. Research on path planning method of mobile robot based on improved bat algorithm[J]. Manufacturing Automation, 2021, 43(6):53-56,62.
[25] 丁元明,侯孟珂. 改进蝙蝠算法的无人机路径规划[J]. 兵器装备工程学报,2023,44(9):26-33.
DING Yuanming, HOU Mengke. UAV path planning based on improved bat algorithm[J]. Journal of Ordnance Equipment Engineering, 2023, 44(9):26-33.
[26] 张瑾,洪莉,戴二壮. 求解带容量和时间窗约束车辆路径问题的改进蝙蝠算法[J]. 计算机工程与科学,2021,43(8):1479-1487.
ZHANG Jin, HONG Li, DAI Erzhuang. An improved bat algorithm for the vehicle routing problem with time windows and capacity constraints[J] Computer Engineering & Science, 2021, 43(8):1479-1487.
[27] TANYILDIZI E, ÖZKAYA G. An improved golden sine algorithm for global optimization problems[J]. Applied Soft Computing, 2018, 68(3):160-175.
[28] DAVIES D L, BOULDIN D W. A cluster separation measure[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1979, 1(2):224-227.
[29] ROUSSEEUW P J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis[J]. Journal of Computational and Applied Mathematics, 1987, 20(7):53-65.

(编辑:陈丽萍)