计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.190600068

基于 Levy 飞行策略的改进樽海鞘群算法

张 严 秦亮曦
广西大学计算机与电子信息学院 南宁 530004
(leelee_yan@163.com)

摘 要 针对樽海鞘群算法(Salp Swarm Algorithm,SSA)在寻优过程中存在的收敛速度较慢、容易陷入局部最优的缺点,提出了一种改进的采用莱维飞行策略的条件化更新的樽海鞘群算法(Levy Flight-based Conditional Updating Salp Swarm Algorithm,LECUSSA),并将其运用于分类算法的特征子集选择过程。首先,利用莱维飞行策略的长短跳跃特点对领导者位置进行随机更新,以增强全局最优的搜索能力;其次,增加对追随者位置的更新条件,让追随者不再盲目地跟随,从而加快收敛速度。在 23 个优化基准函数上对 LECUSSA 算法与其他算法进行了性能比较实验;并把算法运用到支持向量机(Support Vector Machine,SVM)算法的分类特征子集选择中,采用 8 个 UCI 数据集对特征选择后的分类结果进行了性能比较实验。实验结果表明,LECUSSA 具有良好的全局最优搜索能力和较快的收敛速度,利用 LECUSSA 算法进行特征选择后,能够找到最佳分类准确率的特征子集。

关键词:樽海鞘群算法;莱维飞行;条件化更新;特征选择
中图法分类号 TP181

Improved Salp Swarm Algorithm Based on Levy Flight Strategy

ZHANG Yan and QIN Liang-xi
School of Computer,Electronics and Information,Guangxi University,Nanning 530004,China

Abstract Aiming at the shortcomings of slow convergence speed and easy to fall into local optimum in the optimization process of the Salp swarm algorithm (SSA),a Levy Flight-based Conditional Updating Salp Swarm Algorithm (abbreviated as LECUSSA) is proposed and it is used in the feature subset selection of classification algorithm. Firstly,the leader position is updated randomly by using the long and short jump characteristics of Levy Flight strategy,which enhances the global optimal search ability. Secondly,the conditional updating condition to the follower's position is added to make the follower no longer follow blindly,thus accelerating the convergence speed. The performance of LECUSSA algorithm is compared with other algorithms on 23 benchmark functions. The algorithm is applied to the selection of classification feature subset of SVM algorithm,and 8 UCI datasets are used to compare the performance of the classification results after feature selection. The experimental results show that LECUSSA has good global optimal search ability and fast convergence speed. After feature selection using LECUSSA algorithm,the feature subset with the best classification accuracy can be found.

Keywords Salp swarm algorithm,Levy flight,Conditional update,Feature selection

1 引言

近年来,元启发式算法被越来越多的学者研究,其基本思想大多来自人类长期对物理、生物、社会等现象的观察和实践,以及对自然界规律的认识和对自然界生物习性的学习,用数学的方法描述这种规律,进而运用这种规律来解决复杂的问题[1]。

群体智能算法是元启发式算法的一种,包括一系列的算法:粒子群算法(Particle Swarm Optimization,PSO)[2]、萤火虫算法(Firefly Algorithm,FA)[3]、布谷鸟算法(Cuckoo Search Algorithm,CS)[4]、灰狼算法(Grey Wolf Optimizer,GWO)[5]、蜻蜓算法(Dragonfly Algorithm,DA)[6]、鲸鱼算法(Whale Optimization Algorithm,WOA)[7]等。在机器学习蓬勃发展的背景下,群体智能算法也被运用到机器学习的特征选择[8]过程中来寻找合适的特征子集以提升机器学习算法的性能。特征较多会导致计算量的增加和学习准确度的降低,选择适当的特征子集是数据预处理阶段非常重要的工作。

2017 年,Mirjalili 等提出了一种新的群体智能算法,即樽

到稿日期:2019-06-13 返修日期:2019-10-28 本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:广西科技计划(桂科 AB16380260);公益性行业(气象)科研专项(GYHY201406027)
This work was supported by Guangxi Key R&D Project (Guike AB16380260) and Specialized Scientific Research in Public Welfare Industry (Meteorology) (GYHY201406027).
通信作者:秦亮曦(qin_lx@126.com)

海鞘群算法[9]，它是一种模仿海洋中樽海鞘生活习性的新的群体智能优化算法。该算法具有计算量小、简单易懂等特点，提出之后很快被广泛研究和应用。文献[10-11]将 SSA 算法用于实际问题并取得了很好的效果。文献[12-13]对 SSA 算法做了一定的改进，改进后的算法在解决实际工程问题时取得了不错的成效。文献[14]将莱维飞行与 SSA 相结合来解决图像分割问题。值得一提的是，一些研究人员通过有效的形式，如加强形式的 SSA 算法[15]、二元化的 SSA 算法[16]、PSO 和 SSA 结合[17]、混沌序列和 SSA 结合[18]等，对 SSA 算法进行改进，并将其有针对地用在特征选择中。这种改进取得了明显的效果，提升了机器学习算法的性能。

虽然樽海鞘群算法在很多方面都有着很好的效果，但其仍然存在着一定的不足：1）不能更好地实现全局最优的搜索；2）盲目地追随，使得该算法不能快速收敛。针对这两点不足，本文提出了一种基于 Levy 飞行策略条件化更新的樽海鞘群算法 LECUSSA。该算法有两个方面的改进：1）引入了 Levy 飞行策略进行领导者位置的更新，能够实现全局最优的搜索；2）在追随者的位置更新中增加了判别条件，使其进行有偏向的更新。最后，采用 23 个基函数对算法性能指标进行了评测，验证了算法改进策略的有效性。

2　樽海鞘群算法和 Levy 飞行策略

2.1　樽海鞘群算法

樽海鞘群算法是由 Mirjalili 等根据海洋中樽海鞘群觅食的生活习性演化而来的一种群体智能算法[9]。樽海鞘是一类小型远海胶质脊索动物，它们有一个透明的桶状身体和类似水母的组织，靠吸水在水中移动；繁殖期，成群活动的樽海鞘会形成一种链式结构，后面的樽海鞘会紧贴着前面的一个。研究者将这种行为数学化，形成了一种全新的算法——樽海鞘群算法，用于解决优化问题。

2.1.1　种群的初始化

将目标环境定义为一个 $N\times D$ 维的空间，其中 N 表示种群的数量，D 代表空间的维度。每个樽海鞘的位置定义为 $\boldsymbol{X}_i=[X_{i1},X_{i2},X_{i3},\cdots,X_{iD}]$，$i=1,2,3,\cdots,N$；目标位置定义为 $\boldsymbol{F}=[F_1,F_2,F_3,\cdots,F_D]$；各维度搜索的范围上界为 $Ub=[ub_1,ub_2,ub_3,\cdots,ub_D]$，下界为 $Lb=[lb_1,lb_2,lb_3,\cdots,lb_D]$。最后，将初始的种群位置按照式(1)进行随机获取：

$$X_{N\times D}=rand(N,D)\times(Ub-Lb)+Lb \tag{1}$$

在种群中，领导者的每一维的值定义为 X_d^1，追随者的每一维的值定义为 X_d^n，其中 $n=2,3,4,\cdots,N$，d 表示所在的维度。

2.1.2　领导者的位置更新

领导者的位置更新与目标位置有关，且必须有一定的随机性，以发挥它在整个环境搜索中的带头作用，即牵引着追随者向目标运动。领导者的位置更新按照式(2)进行：

$$X_d^1=\begin{cases}F_d+c_1((ub-lb)c_2+lb), & c_3\geqslant 0.5\\ F_d-c_1((ub-lb)c_2+lb), & c_3<0.5\end{cases} \tag{2}$$

其中，F_d 是目标位置 d 维的值；c_1，c_2，c_3 是控制参数。c_2，c_3 是[0，1]之间的随机数，决定领导者位置更新的方向及步长。c_1 叫做收敛因子，用于平衡算法在迭代过程中的收敛速度如式(3)所示：

$$c_1=2\mathrm{e}^{-(4\times l/l_{\max})^2} \tag{3}$$

其中，l 表示当前迭代的次数，$l_{\max}$ 表示最大迭代次数。

2.1.3　追随者的位置更新

追随者是在领导者的基础上进行更新的，不是随机移动的。根据牛顿运动定律，对追随者的速度、加速度、移动位置进行公式化，得到其位置 X_d^n：

$$X_d^n=\frac{1}{2}at^2+v_0t \tag{4}$$

在迭代过程中，t 表示迭代时间；a 表示加速度，且 $a=(v_{\text{final}}-v_0)/t$，$v_0$ 表示初始速度，每次迭代开始时追随者的速度为 0。由于追随者位置的更新只与它前一个樽海鞘的位置有关，因此其速度 $v_{\text{final}}=(X_d^{n-1}-X_d^n)/t$。这个位置更新公式进而可以表示为：

$$X_d^n=\frac{1}{2}(X_d^{n-1}+X_d^n) \tag{5}$$

其中，$n\geqslant 2$，X_d^n 表示第 n 个樽海鞘在维度 d 上的坐标值。

根据以上描述的种群个体更新方式，可以得到樽海鞘群算法，如算法 1 所示。

算法 1　樽海鞘群算法

1. 随机初始化种群中 N 个个体的空间位置坐标；
2. 计算每一个樽海鞘位置的适应度，取最小适应度的位置作为目标位置 F；
3. 根据式(2)更新领导者的位置，根据式(5)更新追随者的位置，然后计算每一个位置的适应度，并将其与当前目标位置的适应度相比较，以最小适应度的位置来更新目标位置；
4. 不停地迭代重复第 3 步，直到迭代次数达到最大时停止循环，最后输出当前的最优适应度和目标位置坐标。

2.2　Levy 飞行策略

自然界中很多生物在不确定的环境中进行觅食时，都采用 Levy 飞行策略作为理想方式。因此，在随机搜索问题中，很多的启发式算法都基于这个策略进行改进并取得了很好的效果[19-22]。由于步长具有随机特点，Levy 飞行策略比布朗运动[23]增长得更快，因此在大规模未知范围内搜索时能够达到比布朗随机运动更好的效果；而且 Levy 飞行是高频短距离探索和低频长距离探索相间的，在大范围内寻找最优解时可以避免陷入局部最优[24]。因此，在很多算法中运用 Levy 飞行能够增加群体分布的多样性，更加快速地寻找到全局最优解[25]。

2.2.1　Levy 分布

Levy 飞行是一种特殊的随机游走策略，它的游走步长分布服从重尾特征[26]的概率分布，称为 Levy 分布，其通常可以近似看作一个简单的幂函数分布 $L(s)\sim|s|^{-1-\beta}$，其中 $0<\beta\leqslant 2$，s 是步长，$L(s)$ 是移动步长 s 的概率。因此 Levy 分布可以表示为：

$$L(s,\gamma,\mu)=\begin{cases}\sqrt{\dfrac{\gamma}{2\pi}}\exp\left[-\dfrac{\gamma}{2(s-\mu)}\right]\dfrac{1}{(s-\mu)^{3/2}}, & 0<\mu<s<\infty\\ 0, & \text{otherwise}\end{cases} \tag{6}$$

其中，$\mu>0$ 是最小步长，γ 是规模参数。明显地，当 $s\to\infty$ 时，式(6)可写成：

$$L(s,\gamma,\mu)\approx\sqrt{\frac{\gamma}{2\pi}}\frac{1}{s^{3/2}} \tag{7}$$

通常情况下，将 $L(s)$ 近似看作：

$$L(s)\rightarrow\frac{\alpha\beta\Gamma(\beta)\sin(\pi\beta/2)}{\pi|s|^{1+\beta}},s\rightarrow\infty \tag{8}$$

其中，Γ 是 gamma 函数。

2.2.2 Levy 步长

在 Mantegna 等的算法[27]中，我们可以将 Levy 飞行步长定义为：

$$s=\frac{u}{|v|^{1/\beta}} \tag{9}$$

其中，s 即为 Levy 飞行路径；u 和 v 是符合正态分布的随机数，$u\sim N(0,\sigma_u^2)$，$v\sim N(0,\sigma_v^2)$。其中的 σ_u，σ_v 由式(10)得到：

$$\begin{cases}\sigma_u=\left\{\dfrac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma[(1+\beta)/2]\beta 2^{(\beta-1)/2}}\right\}^{1/\beta}\\ \sigma_v=1\end{cases} \tag{10}$$

其中，参数 β 的取值范围为 $0<\beta<2$，一般 $\beta=1.5$。

3 算法改进策略

针对樽海鞘群算法中的两点不足，本文分别从领导者的位置更新和追随者的位置更新两个方面进行改进。其中，对领导者的改进策略是增加算法搜索距离的变化，从而提升全局搜索能力；对追随者的改进策略是使其更有针对性地移动，从而加快最优解的搜索速度。

3.1 领导者位置更新改进策略

采用 Levy 飞行随机步长对领导者的位置更新进行改进。Levy 飞行策略使得算法能够在长短距离之间随机变化，利用其中的少数长距离跳跃来避免算法陷入局部最优，增强了全局最优搜索能力。领导者的位置依据式(11)进行更新：

$$X^1=F+c_1[(Ub-Lb)\oplus L(\lambda)+Lb] \tag{11}$$

其中，F 是基础目标位置，$L(\lambda)$ 是 Levy 飞行路径。低频长距离探索可以使算法更好地跳出局部最优，向全局最优搜索，从而可以适当地解决算法易陷入局部最优的问题。

3.2 追随者位置更新改进策略

追随者在原有的樽海鞘群算法中盲目地跟随前一个樽海鞘，这会使得它错过更好的适应度位置。在改进算法 LECUSSA 中，采用有条件的位置更新，首先将前一个樽海鞘的适应度与当前的适应度进行比较，使得新位置更加地偏向适应度较好的一侧。因此追随者的位置按照式(12)所示的改进方式进行更新：

$$X_d^n=r(X_d^{n-1}+X_d^n) \tag{12}$$

其中，r 是位置偏移的系数，其计算式如下：

$$r=\begin{cases}0.5\times rand(0,1), & f(X_d^{n-1})<f(X_d^n)\\ 0.5, & f(X_d^{n-1})=f(X_d^n)\\ 1-0.5\times rand(0,1), & f(X_d^{n-1})>f(X_d^n)\end{cases} \tag{13}$$

其中，f 是适应度函数。

追随者的条件化更新，使得算法能够更加快地向最优解靠近，不至于盲目地进行位置更新，从而加快算法的收敛速度。改进的算法 LECUSSA 如算法 2 所示。

算法 2 改进的算法 LECUSSA

步骤 1 初始化一个规模为 N 的樽海鞘种群，并在(lb, ub)内随机初始化种群的位置；

步骤 2 计算种群中每个樽海鞘的适应度 f 值，将适应度最小(记为 f_{best})的樽海鞘的位置定义为目标位置 F；

步骤 3 按照式(11)更新领导者的位置；

步骤 4 对当前的樽海鞘位置的适应度和前一个樽海鞘位置的适应度进行比较，根据式(13)得出偏移系数 r，再根据式(12)更新跟随者的位置；

步骤 5 根据步骤 2 更新最小适应度 f_{best} 和目标位置 F；

步骤 6 不停地迭代步骤 3－步骤 5，直到达到最大迭代次数后停止，并输出当前的目标位置 F 和最佳适应度 f_{best}。

4 特征选择

本节将 LECUSSA 算法运用到机器学习特征选择中。特征选择是机器学习中数据预处理阶段非常重要的环节，选择代表性的特征能够明显提升机器学习算法的性能(如分类准确率)，因此结合寻优算法的特点，运用元启发式算法能自主地寻找最优解的优势，进行机器学习特征的选择。运用算法 LECUSSA 进行特征选择的实现方式是将樽海鞘群体坐标进行二元化，如[1,1,0,1,0,0,…]，其中数字 1 表示该特征被选中，数字 0 表示该特征被舍去，从而实现了特征选择。

4.1 适应度函数

在基准函数中，适应度值是以每个樽海鞘所在位置坐标为参数的适宜度函数值，而每次迭代的目标是寻找樽海鞘所在位置的最小值。在特征选择中(以分类算法 SVM 为例)，适应度函数定义为寻求分类错误率的最小值，也就是最大化分类准确率：

$$f_n=\text{minimize}(1-accuracy) \tag{14}$$

其中，准确率 $accuracy=\dfrac{n}{N}$，n 表示测试集分类正确的数量，N 表示测试集的总样本数。

4.2 特征选择的寻优算法

在 LECUSSA 算法中，可通过加入实现选择特征的操作来实现特征子集寻优。其中最关键的是对每个樽海鞘的位置坐标进行二元化，实现特征的筛选。设定一个阈值 α，进行二元化操作：

$$X_d^i=\begin{cases}1, & X_d^i\geqslant\alpha\\ 0, & X_d^i<\alpha\end{cases} \tag{15}$$

其中，α 为设定的阈值，其取值为[0,1]之间的随机值；X_d^i 为每个樽海鞘每一维的坐标值。

然后，按照式(16)对数据集 $Data$ 进行筛选：

$$Data_{new}=X\oplus Data \tag{16}$$

其中，$\oplus$ 表示点乘。

特征选择的寻优算法如算法 3 所示。

算法 3 特征选择的寻优算法

步骤 1 初始化一个规模为 N 的樽海鞘种群，在(lb, ub)范围之间随机初始化种群的位置；

步骤 2 根据式(15)进行位置二元化，重新生成樽海鞘坐标值；

步骤 3 根据式(16)进行特征选择，生成选择后的数据集，然后进行机器学习 SVM 分类并计算适应度 f_n，将适应度最小(记为 f_{best})的樽海鞘的位置定义为目标位置 F；

步骤 4　按照式(11)更新领导者的位置；

步骤 5　将当前樽海鞘位置的适应度与前一个樽海鞘位置的适应度进行比较，根据式(13)得出偏移系数 r，再根据式(12)进行跟随者位置的更新；

步骤 6　根据步骤 2 和步骤 3 进行最小适应度 f_{best} 和目标位置 F 的更新；

步骤 7　不停地迭代步骤 4—步骤 6，直到达到最大迭代次数后停止，并输出当前的最佳适应度 f_{best}。

5　仿真实验及结果分析

为了验证改进算法的性能和特征选择的效果，分别进行了实验 1 和实验 2。实验 1 采用 23 个基准函数，将 LECUSSA 与其他的启发式算法进行性能比较；实验 2 则采用 8 个 UCI 机器学习数据集，将 LECUSSA 与 SSA 和 PSO 算法进行特征选择性能的对比。

5.1　实验 1

用 LECUSSA 算法来解决全局最优化问题，并将该算法与其他启发式算法如 SSA，PSO，WOA，GWO，CS，GSA 的性能进行对比。

5.1.1　实验参数的设置

将樽海鞘种群的初始数量设为 30，迭代次数设为 500，取 30 次实验结果的平均值和标准差。23 个基准函数大致可以分成 3 类：单峰值的基准函数，如表 1 所列；多峰值的基准函数，如表 2 所列；固定维度多峰值的基准函数，如表 3 所列。每张表中都列出了基准函数的维度、取值范围和最小值。

表 1　单峰值的基准函数

Table 1　Uni-modal benchmark functions

Function	Dim	Range	f_{min}
$f_1(x)=\sum_{i=1}^{n}x_i^2$	30	[−100,100]	0
$f_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	10	[−10,10]	0
$f_3(x)=\sum_{i=1}^{n}(\sum_{j=1}^{i}x_j)^2$	10	[−100,100]	0
$f_4(x)=\max_i\{\lvert x_i\rvert,1\leqslant i\leqslant n\}$	10	[−100,100]	0
$f_5(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	10	[−30,30]	0
$f_6(x)=\sum_{i=1}^{n}(x_i+0.5)^2$	10	[−100,100]	0
$f_7(x)=\sum_{i=1}^{n}ix_i^4+rand(0,1)$	10	[−1.28,1.28]	0

表 2　多峰值的基准函数

Table 2　Multi-modal benchmark functions

Function	Dim	Range	f_{min}
$f_8(x)=\sum_{i=1}^{n}[x_i^2\sin(\sqrt{\lvert x_i\rvert})]$	10	[−500,500]	$-418.983\times n$
$f_9(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	10	[−5.12,5.12]	0
$f_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$	10	[−32,32]	0
$f_{11}(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	10	[−600,600]	0
$f_{12}(x)=\frac{\pi}{n}\left\{10\sin(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_n-1)^2\right\}+\sum_{i=1}^{n}u(x_i,10,100,4)$	10	[−50,50]	0
$f_{13}(x)=0.1\sin^2(3\pi x_1)+\sum_{i=1}^{n-1}(x_i-1)^2[1+\sin^2(3\pi x_i+1)]+(x_n-1)^2[1+\sin^2(2\pi x_n)]+\sum_{i=1}^{n}u(x_i,5,100,4)$	10	[−50,50]	0

表 3　固定维度多峰值的基准函数

Table 3　Fixed-dimension multi-modal benchmark functions

Function	Dim	Range	f_{min}
$f_{14}(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$	2	[−65,65]	1
$f_{15}(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_i(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}\right]^2$	4	[−5,5]	0.000 30
$f_{16}(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_1^4$	2	[−5,5]	−1.031 6
$f_{17}(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	2	[−5,5]	0.398
$f_{18}(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]\times[30+(2x_1-3x_2)^2\times(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	2	[−2,2]	3
$f_{19}(x)=-\sum_{i=1}^{4}c_i\exp\left(\sum_{j=1}^{3}a_{ij}(x_j-p_{ij})^2\right)$	3	[0,1]	−3.86
$f_{20}(x)=-\sum_{i=1}^{4}c_i\exp\left(\sum_{j=1}^{6}a_{ij}(x_j-p_{ij})^2\right)$	6	[0,1]	−3.32
$f_{21}(x)=-\sum_{i=1}^{5}[(x-a_i)(x-a_i)^{\mathrm{T}}+c_i]^{-1}$	4	[0,10]	−10.153 2
$f_{22}(x)=-\sum_{i=1}^{7}[(x-a_i)(x-a_i)^{\mathrm{T}}+c_i]^{-1}$	4	[0,10]	−10.402 8
$f_{23}(x)=-\sum_{i=1}^{10}[(x-a_i)(x-a_i)^{\mathrm{T}}+c_i]^{-1}$	4	[0,10]	−10.536 4

5.1.2 实验1的结果及分析

23个基准函数的实验结果如表4－表6所列，所有结果均为30次实验结果的平均值和标准差，加粗字体表示与其他算法对比时能寻到的最小值。

表4 单峰值基准函数的实验结果

Table 4 Optimization resultson uni-modal benchmark functions

f	Index	LECUSSA	SSA	PSO	WOA	GWO	CS	GSA
$f_1(x)$	Mean	2.65×10^{-13}	3.29×10^{-7}	1.31×10^{-5}	**1.41×10^{-30}**	1.27×10^{-27}	5.78×10^{-3}	2.53×10^{-16}
	Std	3.94×10^{-13}	5.92×10^{-7}	2.39×10^{-5}	4.91×10^{-30}	3.11×10^{-27}	2.41×10^{-3}	9.67×10^{-17}
$f_2(x)$	Mean	5.36×10^{-8}	1.9111	0.0076	**1.06×10^{-21}**	8.52×10^{-17}	0.2080	5.57×10^{-2}
	Std	5.76×10^{-8}	1.6142	0.0262	2.39×10^{-21}	6.62×10^{-17}	3.17×10^{-2}	0.19407
$f_3(x)$	Mean	**1.30×10^{-12}**	1.50×10^{-3}	2.13×10^{-3}	5.39×10^{-7}	2.43×10^{-5}	2.63×10^{-1}	8.97×10^{-2}
	Std	3.13×10^{-12}	707.0529	878.7605	2.93×10^{-6}	8.14×10^{-5}	2.97×10^{-2}	318.96
$f_4(x)$	Mean	**1.10×10^{-7}**	2.44×10^{-5}	16.4104	0.072581	7.69×10^{-7}	1.43×10^{-5}	7.35
	Std	1.11×10^{-7}	1.89×10^{-5}	4.3603	0.39747	6.51×10^{-7}	4.83×10^{-6}	1.74145
$f_5(x)$	Mean	8.9492	136.5676	106.6465	27.86558	27.1786	**0.008**	67.543
	Std	4.01×10^{-2}	1.54×10^{-2}	104.3333	0.763626	0.814	0.054	62.225
$f_6(x)$	Mean	1.03×10^{-3}	1.72×10^{-7}	5.59×10^{-5}	3.116266	0.70757	6.17×10^{-4}	**2.50×10^{-16}**
	Std	1.55×10^{-3}	2.44×10^{-7}	1.42×10^{-4}	0.532429	0.3632	2.8×10^{-5}	1.74×10^{-16}
$f_7(x)$	Mean	**1.01×10^{-4}**	0.169	6.05×10^{-2}	0.001425	1.72×10^{-3}	0.02855	8.94×10^{-2}
	Std	8.86×10^{-5}	0.0686	0.0206	0.001149	1.10×10^{-3}	0.001277	0.04339

表5 多峰值基准函数的实验结果

Table 5 Optimization results on multi-modal benchmark functions

f	Index	LECUSSA	SSA	PSO	WOA	GWO	CS	GSA
$f_8(x)$	Mean	−3005.39	−7460	**−9347.8**	−5080.7	−5776.12	−2128.91	−2821.07
	Std	0.131	634.6745	754.483	695.796	682.0101	0.0084	493.037
$f_9(x)$	Mean	2.98×10^{-14}	55.4523	58.0837	**0**	3.1496	0.246	25.968
	Std	5.26×10^{-14}	18.2751	13.1635	0	4.0294	0.0018	7.47
$f_{10}(x)$	Mean	8.75×10^{-9}	2.8401	0.9353	7.4043	**1.03×10^{-13}**	4.1×10^{-10}	0.0621
	Std	8.33×10^{-8}	0.6581	0.9909	9.89757	1.60×10^{-14}	5.2×10^{-9}	0.2363
$f_{11}(x)$	Mean	**4.11×10^{-13}**	0.2291	1.9×10^{-2}	0.00028	6.08×10^{-3}	0.1852	27.702
	Std	8.46×10^{-13}	0.1295	0.0232	0.00158	0.0111	0.039	5.0403
$f_{12}(x)$	Mean	**1.90×10^{-4}**	6.8274	0.9099	0.33967	0.03669	0.01258	1.79962
	Std	2.04×10^{-4}	2.7192	1.0678	0.21486	0.0175	4.1×10^{-9}	0.9511
$f_{13}(x)$	Mean	0.93277	21.3116	**0.119**	1.88901	0.62239	0.485	8.8991
	Std	0.13204	16.9894	0.2396	0.26608	0.2837	6.8×10^{-8}	7.1262

表6 固定维度多峰值基准函数的实验结果

Table 6 Optimization results on fixed-dimension multi-modal benchmark functions

f	Index	LECUSSA	SSA	PSO	WOA	GWO	CS	GSA
$f_{14}(x)$	Mean	**0.998**	1.1301	**0.998**	2.111973	4.917	1.4236	5.8598
	Std	7.15×10^{-11}	0.5659	0	2.498594	4.125	$1.3E\times10^{-2}$	3.83
$f_{15}(x)$	Mean	0.00167	0.0027	3.20×10^{-3}	0.000572	5.08×10^{-3}	**5.0×10^{-4}**	0.003673
	Std	0.0010	0.00543	0.0071	0.000324	0.0086	1.11×10^{-4}	1.65×10^{-3}
$f_{16}(x)$	Mean	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**
	Std	1.12×10^{-5}	5.78×10^{-14}	6.71×10^{-16}	$4.2E\times10^{-7}$	2.64×10^{-8}	1.49×10^{-8}	4.88×10^{-16}
$f_{17}(x)$	Mean	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**
	Std	7.34×10^{-5}	1.26×10^{-14}	0	2.7×10^{-5}	8.86×10^{-7}	3.24×10^{-6}	0
$f_{18}(x)$	Mean	3.0005	**3**	**3**	**3**	**3**	3.0014	**3**
	Std	0.00149	2.66×10^{-13}	1.26×10^{-15}	4.22×10^{-15}	4.27×10^{-5}	0.00258	4.17×10^{-15}
$f_{19}(x)$	Mean	−3.8523	**−3.8628**	−3.8625	−3.85616	−3.861	−3.268	−3.86278
	Std	0.019678	1.13×10^{-11}	0.0014	0.002706	0.0023	1.85×10^{-5}	2.29×10^{-15}
$f_{20}(x)$	Mean	−3.2728	−3.22526	−3.2518	−2.98105	−3.2568	**−3.32185**	−3.31778
	Std	0.079	0.05772	0.1086	0.376653	0.0885	7.21E−03	0.02308
$f_{21}(x)$	Mean	**−10.1523**	−7.33	−6.2948	−7.04918	−9.3955	−9.728	−5.95512
	Std	2.82×10^{-7}	3.3158	2.9249	3.629551	2	0.2881	3.73708
$f_{22}(x)$	Mean	**−10.4028**	−8.48022	−7.0398	−8.18178	−10.2241	−9.873	−9.68447
	Std	2.47×10^{-7}	3.094025	3.5751	3.829202	0.9701	0.32034	2.014
$f_{23}(x)$	Mean	**−10.5364**	−8.64195	−7.7011	−9.34238	−10.0838	−9.7822	**−10.5364**
	Std	5.23×10^{-8}	3.1316	3.6189	2.414737	1.7514	0.5002	2.6×10^{-15}

实验结果表明，改进的算法LECUSSA在很多基函数中均能达到最低搜索值，在基函数 $f_3(x)$，$f_7(x)$，$f_{11}(x)$ 和 $f_{12}(x)$ 中明显优于算法SSA，相比于其他算法也存在明显的优势；在 $f_{21}(x)$，$f_{22}(x)$ 和 $f_{23}(x)$ 这3个函数中不仅能够寻到最小值，而且收敛稳定性亦优于其他算法。

图1－图6给出了LECUSSA与其他4种算法对部分基准函数进行优化过程中的适应度曲线。由图1－图6的优化曲线可明显地看出LECUSSA算法的寻优性能优于其他4种算法。由图5、图6可以看出，在同时寻到最优解的情况下，LECUSSA算法在收敛速度上优于其他4种算法。LECUSSA算法采取的位置更新改进策略不仅加快了算法的收敛速度，也增强了算法对最优解的搜索能力。

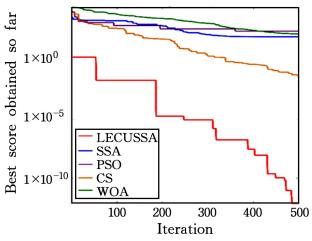

图 1　f_3 函数优化曲线

Fig. 1　f_3 optimization curves

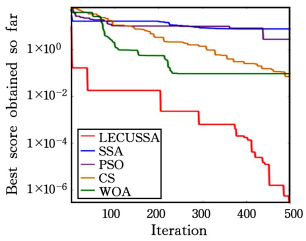

图 2　f_4 函数优化曲线

Fig. 2　f_4 optimization curves

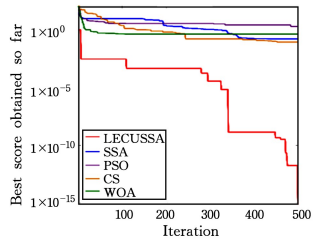

图 3　f_{11} 函数优化曲线

Fig. 3　f_{11} optimization curves

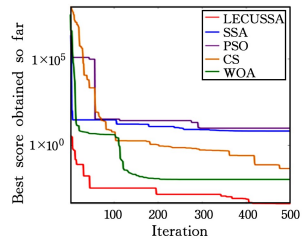

图 4　f_{12} 函数优化曲线

Fig. 4　f_{12} optimization curves

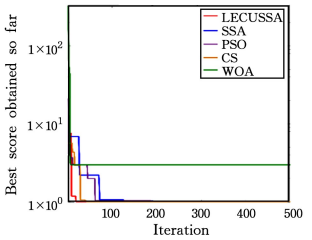

图 5　f_{14} 函数优化曲线

Fig. 5　f_{14} optimization curves

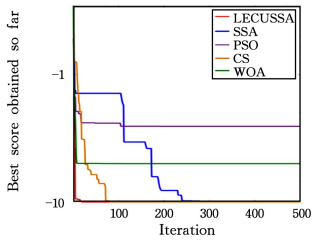

图 6　f_{21} 函数优化曲线

Fig. 6　f_{21} optimization curves

5.2　实验 2

实验 2 将 LECUSSA 算法运用到 SVM 分类算法中作为预处理阶段的特征选择算法，以提高分类算法的性能，进而提高分类准确率。

5.2.1　实验参数设置

LECUSSA 算法的参数设置如下：初始种群数为 30，最大迭代次数为 50，种群个体位置坐标取值范围为[0，1]。将数据集分为训练集和测试集，采用 10 折交叉验证方法进行实验，输出 10 次实验的平均准确率、标准差、最佳的实验准确率和不进行特征选择的原始准确率。实验数据集选自 UCI 机器学习数据集，共 8 个，表 7 列出了这 8 个数据集具有的特性。

表 7　UCI 数据集的特性

Table 7　Characters of UCI datasets

ID	Name	Number of samples	Number of features
D_1	Zoo	101	16
D_2	Wine	178	13
D_3	Iris	150	4
D_4	Parkinsons	197	22
D_5	Lymphography	148	18
D_6	Sonar	208	60
D_7	Statlog (Heart)	270	13
D_8	Glass Identification	214	10

5.2.2　实验 2 的结果及分析

表 8 列出了经过算法 LECUSSA，SSA，PSO 进行特征选择之后，采用 SVM 算法进行分类得到的结果。其中呈现了 3 种算法进行降维后的 10 次实验的平均特征数、标准差、平均准确率和最佳准确率。

表 8　特征选择实验结果

Table 8　Feature selection results

ID	Original Accuracy	LECUSSA				SSA				PSO			
		Mean of Feature	Std	Mean	Best	Mean of Feature	Std	Mean	Best	Mean of Feature	Std	Mean	Best
D_1	0.867	7.7	0.026	0.957	1.000	7.1	0.024	0.910	0.967	8.0	0.011	0.0970	1.000
D_2	0.963	8.4	0	1.000	1.000	10.4	0	1.000	1.000	6.6	0	1.000	1.000
D_3	0.956	2.9	0.016	0.989	1.000	3	1.17×10^{-16}	1.000	1.000	2.4	1.17×10^{-16}	1.000	1.000
D_4	0.881	17.2	0.027	0.935	**0.983**	17.7	0	0.962	0.966	14.5	0.014	0.964	0.967
D_5	0.386	6.5	0.040	0.473	**0.558**	14	0.021	0.509	0.523	7.5	0.014	0.535	**0.558**
D_6	0.810	25.2	0.020	0.925	**1.000**	17.7	0.020	0.890	0.919	18.5	0.020	0.934	0.952
D_7	0.802	5.9	0.034	0.869	**0.914**	6.1	0.017	0.847	0.864	5.3	0.019	0.870	0.889
D_8	0.892	8.2	0.038	0.944	**0.985**	5.7	0.008	0.836	0.937	5.6	0.005	0.937	0.937

在 8 个 UCI 数据集中的平均准确率都比不做特征选择的原始特征集的分类准确率高很多，LECUSSA 算法在这 8 个数据集中的平均准确率并不都占优势，而且从标准差的结果可以看出，其迭代选择的结果没有其他两种算法选择结果的稳定性好，这是由于算法中使用了 Levy 飞行的更新方式而使得搜索的跨度比较大所导致的结果；但是 LECUSSA 算法在数据集 D_4，D_6，D_7，D_8 中的最佳准确率明显高于其他两种算法的结果，在其他的数据集中也能够达到与其他两种算法同样的最佳准确率。可见，在特征选择中，LECUSSA 算法明显能选择具有较好分类效果的特征子集，只是在稳定性上有待改善。

结束语　本文提出了一种基于 Levy 飞行和有条件更新的改进樽海鞘群算法 LECUSSA。在 23 个基准函数上进行的性能比较实验表明，LECUSSA 算法采用的改进策略是有效的。在 8 个 UCI 数据集上进行的仿真实验表明，将改进算法 LECUSSA 与分类算法结合是有效的，能较好地提高分类准确率。未来，我们可以尝试新的算法改进思路，并将启发式算法和不同的机器学习算法相结合进行特征选择，以提升机器学习算法的性能。

参 考 文 献

[1]　WEBB B. Swarm Intelligence: From Natural to Artificial Systems[J]. Connection Science, 2002, 14(2): 163-164.

[2]　KENNEDY J, EBERHART R. Particle swarm optimization

[C]// Proceedings of the IEEE International Conference on Neural Networks. 1995:1942-1948.
[3] YANG X S. Firefly algorithms for multimodal optimization [C]// International Symposium on Stochastic Algorithms. Berlin:Springer,2009:169-178.
[4] YANG X S,DEB S. Cuckoo search via Lévy flights[C]// 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC). IEEE Press,2009:210-214.
[5] MIRJALILI S,MIRJALILI S M,LEWIS A. Grey Wolf Optimizer[J]. Advances in Engineering Software,2014,69:46-61.
[6] MIRJALILI S. Dragonfly algorithm:a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems[J]. Neural Computing and Applications,2016,27(4):1053-1073.
[7] MIRJALILI S,LEWIS A. The Whale Optimization Algorithm [J]. Advances in Engineering Software,2016,95:51-67.
[8] LIU H,MOTODA H. Feature selection for knowledge discovery and data mining[M]. Springer Science & Business Media,2012:73-95.
[9] MIRJALILI S,GANDOMI A H,MIRJALILI S Z,et al. Salp Swarm Algorithm:A bio-inspired optimizer for engineering design problems[J]. Advances in Engineering Software, 2017, 114:163-191.
[10] CHEN T,WANG M X,HUANG X S. Passive TDOA location based on Bohai Sheath Swarm Algorithms [J]. Chinese Journal of Electronics and Information,2018,40(7):72-78.
[11] HUSSIEN A G,HASSANIEN A E,HOUSSEIN E H. Swarming behaviour of salps algorithm for predicting chemical compound activities[C]// 2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS). IEEE Press,2017:315-320.
[12] WANG M Q,WANG Y,JI Z C. PMSM multi-parameter identification based on improved salp swarm algorithm [J]. Chinese Journal of System Simulation,2018(11):4284-4291.
[13] QAIS M H,HASANIEN H M,ALGHUWAINEM S. Enhanced salp swarm algorithm:Application to variable speed wind generators[J]. Engineering Applications of Artificial Intelligence, 2019,80:82-96.
[14] XING Z K,JIA H M,SONG W L. Multi-threshold image segmentation based on Levy's Flying Bowl Sheath swarm optimization algorithm [J/OL]. Chinese Journal of Automation. [2019-08-10]. https://doi.org/10.16383/j.aas.c180140.
[15] HEGAZY A E,MAKHLOUF M A,EL-TAWEL G S. Improved salp swarm algorithm for feature selection[J]. Journal of King Saud University-Computer and Information Sciences,2020,32(3):355-344.
[16] FARIS H,MAFARJA M M,HEIDARI A A,et al. An efficient binary salp swarm algorithm with crossover scheme for feature selection problems[J]. Knowledge-Based Systems, 2018, 154:43-67.
[17] IBRAHIM R A,EWEES A A,OLIVA D,et al. Improved salp swarm algorithm based on particle swarm optimization for feature selection[J]. Journal of Ambient Intelligence and Humanized Computing,2019,10(8):3155-3169.
[18] SAYED G I,KHORIBA G,HAGGAG M H. A novel chaotic salp swarm algorithm for global optimization and feature selection[J]. Applied Intelligence,2018,48(10):3462-3481.
[19] YAN X F,YE D Y. An improved flora foraging algorithm based on Levy flight [J]. Computer system applications,2015,24 (3):124-132.
[20] HÜSEYIN H,HARUN U. A novel particle swarm optimization algorithm with LeÜvy flight[J]. Applied Soft Computing,2014, 23:333-345.
[21] AYDOĞDU Î,AKIN A,SAKA M P. Design optimization of real world steel space frames using artificial bee colony algorithm with Levy flight distribution[J]. Advances in Engineering Software,2016,92:1-14.
[22] LIU C P,YE C M. Bat algorithm with Levy flight characteristics [J]. Chinese Journal of Intelligent Systems, 2013, 8(3):240-246.
[23] BIAGINI F. Stochastic calculus for fractional Brownian motion and applications[M]. London:Springer,2008:5-20.
[24] ZHU X E,HAO X,XIA S R. Levy flight-based feature selection algorithm [J]. Chinese Journal of Zhejiang University (Engineering Edition),2013(4):638-643.
[25] EDWARDS A M,PHILLIPS R A,WATKINS N W,et al. Revisiting Lévyflight search patterns of wandering albatrosses, bumblebees and deer [J]. Nature,2007,449(7165):1044-1048.
[26] EMBRECHTS P,KLÜPPELBERG C,MIKOSCH T. Modelling extremal events:for insurance and finance[M]// Science & Business Media. Springer,2013:371-403.
[27] MANTEGNA R N. Fast,accurate algorithm for numerical simulation of Lévy stable stochastic processes[J]. Physical Review. E,1994,49(5):4677-4689.

ZHANG Yan, born in 1991, postgraduate. His main research interests include data mining, machine learning and deep learning.

QIN Liang-xi, born in 1963, Ph.D, professor, is a member of China Computer Federation. His main research interests include data mining, decision rough set and deep learning. etc.