[文章编号] 1003－4684(2021)04-0001-03

基于改进的蜻蜓算法的特征选择

刘东强，陈宏伟

（湖北工业大学计算机学院，湖北 武汉 430068）

[摘 要] 蜻蜓算法是一种近年提出的元启发式优化算法，它主要是模拟自然界中蜻蜓的捕食和迁徙行为。原始的蜻蜓算法跟其他许多群体智能优化算法一样，存在着自身的缺陷，容易陷入局部最优，并且收敛速度较慢。为了提高蜻蜓优化算法的性能，在算法种群初始化阶段引入混沌映射策略，提高了初代种群的质量，并且将原蜻蜓算法的线性惯性权重做了非线性改进，提高了算法的收敛速度，最后运用于特征选择来检验其实际效果。实验结果表明，改进后的蜻蜓算法比原算法的效果更好。

[关键词] 蜻蜓算法；混沌映射；惯性权重；特征选择

[中图分类号] TP399 [文献标识码] A

近些年来，人们根据生物种群的各种行为，相继提出了各种各样的群体智能优化算法，并取得了诸多成果。例如遗传算法[1]，粒子群优化算法[2]，人工蜂群优化算法[3]，萤火虫算法[4]等。这些算法被研究者们很好的运用到了社会的各个领域。蜻蜓算法也是一种新颖的元启发式优化算法，它模拟蜻蜓的群体行为，是由 Seyedali Mirjalili 为解决连续优化问题在 2015 年提出的。蜻蜓算法被许多学者研究与运用，如崔学婷等人提出了一种混合改进蜻蜓算法并将其应用于特征选择[5]；文献[6]利用二进制蜻蜓算法设计了一种特征选择包装器，提高了特征选择分类效果；王万良等人为合理应用生产制造中产生的数据，分析和挖掘出数据中的关键特征和潜在信息，以帮助企业提高效率、降低成本，提出一种改进的蜻蜓算法，并将其用于特征选择[7]。为了提高蜻蜓算法的效率与精度，本文提出了一种改进的蜻蜓算法并运用于特征选择。文章在算法的种群初始化过程中引入混沌策略，并将原来的线性权重做了一个非线性调整，最终用具体的实验数据验证了改进后的蜻蜓算法的分类精度更好。

1 蜻蜓算法

蜻蜓算法是一种新的智群优化算法，该算法具有很强的分类搜索能力和迭代优化能力，已在各个领域得到了很好的应用。蜻蜓算法的主要灵感源于自然界中蜻蜓的群体行为[8]。蜻蜓有两个重要的群体行为：狩猎和迁徙。前者称为静态群体，后者称为动态群体。在静态群中，蜻蜓会成群结队，在一个小区域内来回飞行，以捕猎其他飞行的猎物，例如蝴蝶和蚊子[9]。在动态群中，大量的蜻蜓使群在一个方向上长距离迁移[10]。

基于蜻蜓的群体行为，建立了 5 个数学模型：

1）分离行为（S_i）计算如下：

$$S_i = -\sum_{j=1}^{n} X - X_j$$

其中 X 是当前个体的位置，X_j 表示第 j 个相邻个体的位置，n 是相邻个体的数量。

2）对齐行为（A_i）计算如下：

$$A_i = \frac{\sum_{j=1}^{n} V_j}{n}$$

其中 V_j 表示第 j 个相邻个体的速度，n 是相邻个体的数量。

3）聚集行为（C_i）计算如下：

$$C_i = \frac{\sum_{j=1}^{n} X_j}{n} - X$$

其中 X 是当前个体的位置，N 是邻域的数目，X_j 表示第 j 个邻域个体的位置。

4）觅食行为（F_i）计算如下：

$$F_i = X^+ - X$$

其中 X 是当前个体的位置，X^+ 显示食物来源的位置。

5）避敌行为（E_i）计算如下：

[收稿日期] 2021－05－09
[基金项目] 国家自然科学基金项目(61772180)
[第一作者] 刘东强(1994－)，男，湖北恩施人，湖北工业大学硕士研究生，研究方向为大数据
[通信作者] 陈宏伟(1975－)，男，湖北武汉人，工学博士，湖北工业大学教授，研究方向为大数据

$$E_i = X^- + X$$

其中 X 是当前个人的位置，X^- 表示敌人的位置。

在进行蜻蜓个体位置更新时，首先计算步进矢量(ΔX)，步进矢量显示了蜻蜓的运动方向，并定义如下：

$$\Delta X_{t+1} = (s S_i + a A_i + c C_i + f F_i + e E_i) + w \Delta X_t \tag{1}$$

其中 s、a、c、f 和 e 为各自所对应的权重，S_i、A_i 和 C_i 分别表示个体的分离、对齐和凝聚力，F_i 表示食物的吸引力，E_i 表示敌人的影响力，w 为惯性权重，t 为迭代计数器。

计算步进矢量后，位置矢量的计算如下：

$$X_{t+1} = X_t + \Delta X_{t+1}$$

2　改进的蜻蜓算法

在本文中，提出了一种基于混沌理论的新的蜻蜓优化算法。普通的蜻蜓算法和其他许多智能优化算法一样，在迭代的过程中，很容易陷入局部最优的陷阱，而真正需要的是全局最优值。针对这一缺陷，本文在种群初始化的过程中加入混沌理论，得到覆盖面更广的初始种群。为了提高算法的收敛速度，对原蜻蜓算法中的惯性权重 w 做了一个非线性的调整，大大提高了算法效率与精度。算法的流程图见图 1。

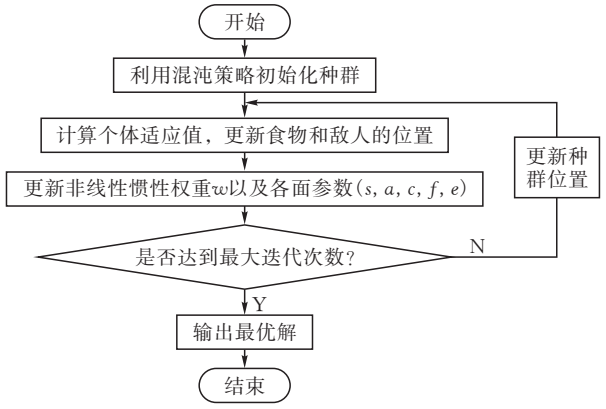

图 1　算法流程图

2.1　基于混沌策略初始化种群

本文中，为了提高初代蜻蜓种群的多样性，而不影响得到的结果，并为算法后期的全局搜索与迭代奠定良好的初始种群基础。本文在种群初始化的过程中，引用 Tent 混沌序列的逆混沌映射策略，得到一个品质更优的初代蜻蜓种群。

在本算法中，每个蜻蜓个体的维度为 d，在初始化种群的时候，本文首先生成一个 d 维的蜻蜓个体(i=1,2,…,n)，然后把生成的个体通过混沌映射，转化到另外一个解集中去，进而生成一个新的个体 CX_{id} (i=1,2,…,n)，计算方法如下：

$$\begin{cases} X_{id} = X_{mind} + \mathrm{rand}(X_{maxd} - X_{mind}) \\ CX_{id} = X_{mind} + X_{maxd} - X_{id} \end{cases}$$

式中 X_{id} 是第 i 个蜻蜓个体的第 d 维上的值，X_{mind} 和 X_{maxd} 分别是取值的下界和上界，种群 $CX = \{CX_i, i=1,2,\cdots,n\}$ 是通过混沌映射由种群 $X = \{X_i, i=1,2,\cdots,n\}$ 变化得到。首先把 X 和 CX 两个种群合并成一个新的蜻蜓种群，其个体数为 $2n$，然后再计算每个蜻蜓个体的适应值，并对得到的所有适应值进行排序，最后取前 n 个适应值所对应的蜻蜓个体来组成蜻蜓算法的初代种群。

2.2　惯性权重 w 的非线性改变

在原蜻蜓算法中，式(1)中的惯性权重 w 是一个线性变化的值，其计算方法见式(2)。在实际情况中，随着算法迭代次数的增加，算法收敛的速度会慢慢降低，是呈曲线下降的。算法中惯性权重 w 收敛的速度和算法整体的收敛速度不一致，这就会降低算法的收敛速度，从而影响算法的整体效率。

$$w = 0.9 - t(0.5/t_{\max}) \tag{2}$$

式(2)中 t 为当前的迭代次数，$t_{\max}$ 为开始设置的总迭代次数。从公式(2)中可以看出，随着迭代次数的增加，惯性权重 w 会呈直线趋势变小。在本文中，引用了指数递减策略，通过下面公式对惯性权重进行一个非线性的调整改变：

$$w = w_{\min} + |w_{\max} - w_{\min}| \exp\left(\frac{-\alpha t}{T}\right)$$

式中 $w_{\max}$ 和 $w_{\min}$ 分别为公式(2)中惯性权重 w 的最大值和最小值：$w_{\max}=0.9$，$w_{\min}=0.4$。其中 α 为调节系数，本文中取 $\alpha=4$，t 和 T 分别为当前迭代次数和最大迭代次数。改进后的惯性权重 w 和原惯性权重 w 的对比图如图 2 所示，从图中可以看出改进后的 w 是呈曲线下降的，相对于原惯性权重 w，其开始的变化速度很快，随着迭代次数的增加，变化的速度逐渐变慢，这能更好地适应算法的整体收敛速度。

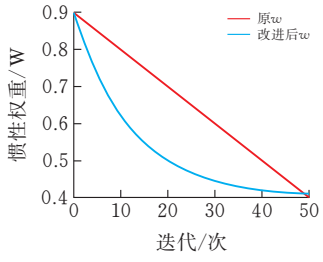

图 2　权重对比图

2.3　特征选择

特征选择是一种二进制优化问题，个体的位置都是由 0 或者 1 来表示。蜻蜓算法能够很好的解决连续优化问题，它通过将步长向量增加到位置向量的方法来改变个体的位置，但在二进制搜索空间中，

种群个体的位置由 0 和 1 表示，此方法就不适用了。根据先前 Mirjalili 和 Lewis 的研究[11]。可以通过传递函数将连续优化算法转换成二进制算法，传递函数一般分为两种：S 型和 V 型。本文采用 V 型传递函数，将原连续优化算法转换成了用于特征选择的二进制优化算法。

3　实验结果与分析

在实验中，本文先将改进后的二进制蜻蜓算法（CBDA）与原始的二进制蜻蜓算法（BDA）进行了比较，使用的数据集是两个 UCI 数据集：Wine 和 Sonar。本实验在 Windows10 操作系统下进行，电脑的相关配置：操作系统 Windows 10，64 bit，处理器 Intel(R) Core(TM) i5-7200U，内存 16 G，编程语言 Python3.5.2。种群的规模都设置为 30，迭代次数都设置为 50。通过比较四种算法得到的最优个体的适应值来判断算法对于文本分类中特征选择的效果。

实验结果见图 3、图 4，从图中可以看出：改进后的二进制蜻蜓算法的收敛速度明显要好于原算法。

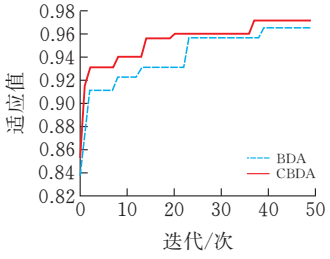

图 3　在 Wine 中函数适应度值

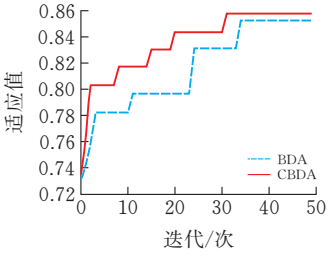

图 4　在 Sonar 中函数适应度值

为验证改进的蜻蜓算法性能，本文还将改进的二进制蜻蜓算法（CBDA）和传统的二进制遗传算法（BGA）、二进制粒子群算法（BPSO）进行比较。从 30 次运行结果中获取每种算法的平均分类精度，然后相互之间进行比较，得到表 1 的结果，从中可知改进后的蜻蜓算法的分类精度要好于其他算法。

表 1　平均分类精度

数据集	CBDA	BGA	BPSO
Wine	0.975	0.963	0.972
Sonar	0.861	0.817	0.791

4　结论

在本文中，将混沌映射和惯性权重的非线性改变运用到蜻蜓算法中，并将改进后的蜻蜓算法运用于特征选择来检验其效果。通过实验结果可以看出，改进后的蜻蜓算法的收敛速度相对于原算法得到了提高，而且相对于传统的优化算法，其分类精度也更高。

[参　考　文　献]

[1] 刘成锴，王斌君，吴勇.基于遗传算法的文本特征选择[J].科学技术与工程，2019，19(33)：302-307.

[2] 袁剑锋.基于改进二进制粒子群优化的特征选择[J].新型工业化，2020，10(7)：21-22.

[3] 韦宣.混合型人工蜂群优化算法及其应用研究[D].镇江：江苏大学，2020.

[4] 卓宏明，陈倩清.基于自适应萤火虫算法的液压缸优化设计[J].数学的实践与认识，2020，50(15)：89-95.

[5] CUI X，LI Y，FAN J，et al. A hybrid improved dragonfly algorithm for feature selection[J]. IEEE Access，2020，8：155619-155629.

[6] MAFARJA M M，ELEYAN D，JABER I，et al. Binary dragonfly algorithm for feature selection[C]. 2017 International Conference on New Trends in Computing Sciences (ICTCS)，2017：12-17.

[7] 王万良，朱凯莉，李伟琨，等.改进蜻蜓算法及其在特征选择中的应用[J].计算机集成制造系统，2020，26(08)：2124-2132.

[8] SEYEDALI M. 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.

[9] WIKELSKI M，MOSKOWITZ D，ADELMAN JAMES S，et al. Simple rules guide dragonfly migration[J]. Biology Letters，2006，2(3)：325-329.

[10] ROBERT W，RUSSELL，MICHAEL L MAY，et al. Fitzpatrick. Massive swarm migrations of dragonflies (odonata) in eastern north america[J]. The American Midland Naturalist，1998，140(2)：325-342.

[11] SEYEDALI M，ANDREW L. S-shaped versus v-shaped transfer functions for binary particle swarm optimization [J]. Swarm and Evolutionary Computation，2013，9：1-14.

（下转第 41 页）