

# 基于粗糙集和果蝇优化算法的特征选择方法

方波 陈红梅 王生武

(西南交通大学信息科学与技术学院 成都 611756)

(西南交通大学云计算与智能技术高校重点实验室 成都 611756)

**摘要** 特征选择是模式识别领域重要的数据预处理步骤之一,旨在从原始特征集中选出最有效的特征子集使得给定评价准则达到最优。为此,文中提出了一种基于粗糙集和果蝇优化算法的特征选择方法。该方法基于一种新的双策略进化果蝇优化算法进行特征子集的迭代寻优,并结合粗糙集属性依赖度和属性重要性构造适应度函数对所选特征子集进行评估,既可以在全局范围内尽可能多地搜索出重要的特征,又能选出对决策最具有贡献的有效特征子集。在 UCI 数据集上的实验结果表明,提出的特征选择方法可以有效地搜索出具有最少信息损失的特征子集,并达到较高的分类精度。

**关键词** 粗糙集,果蝇优化算法,双策略进化,属性依赖度,属性重要性

**中图分类号** TP301.6 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2019.07.025

## Feature Selection Algorithm Based on Rough Sets and Fruit Fly Optimization

FANG Bo CHEN Hong-mei WANG Sheng-wu

(School of Information Science and Technology, Southwest Jiaotong University, Chengdu 611756, China)

(Key Laboratory of Cloud Computing and Intelligent Technology, Southwest Jiaotong University, Chengdu 611756, China)

**Abstract** Feature selection is one of the most important data preprocessing steps in the field of pattern recognition, aiming at searching the most effective subset with the best value of evaluation function from original data set. This paper proposed a new feature selection strategy based on the rough set theory and fruit fly optimization algorithm. The novel double strategies evolutionary fruit fly optimization algorithm(DSEFOA) is used to search feature subset and execute the iterative optimization. Specially, the selected feature subset is evaluated by the fitness function constructed by attribute dependency and attribute importance simultaneously, which aims at searching important features as many as possible in feature space and further selecting effective feature subset with the most contribution to the decision. Experimental results on UCI datasets show that the proposed feature selection method can effectively search the feature subset with the minimum information loss and achieve high classification accuracy.

**Keywords** Rough sets, Fruit fly optimization algorithm, Double strategies evolutionary, Attribute dependency, Attribute importance

## 1 引言

在当前大数据迅猛发展的环境下,特征选择作为一种有效的数据处理方式,成为机器学习、模式识别和数据挖掘等领域的一个热门研究课题,受到了广泛的关注和重视。特征选择通过去除不相关、冗余或存在噪声的特征属性,从数据的原始特征集中选择一些使评价准则最优的特征,以达到对特征空间降维的目的,进而降低数据存储和处理的成本<sup>[1-2]</sup>。

根据与分类算法相结合的方式,特征选择方法可分为过滤式(Filter)方法、封装式(Wrapper)方法和嵌入式(Embedded)方法<sup>[3-4]</sup>。过滤式方法旨在构造一个独立于学习算法之

外的评价指标来对特征进行评分,其评价准则与分类器无关,该方法具有较好的通用性和搜索速率,但效果一般,选出的特征子集未必最小<sup>[5]</sup>;封装式方法直接使用分类器的分类性能来评估所选特征的分类能力,该方法能取得较高的分类精度,但由于其计算量依赖所选择的分类学习算法,往往计算复杂度高,不适合直接应用于大数据降维<sup>[6]</sup>;而嵌入式的方法在学习分类器的同时进行特征选择,通常与特定的分类任务绑定,不具有通用性,不能单独作为统一的大数据预处理方式<sup>[7]</sup>。

粗糙集理论由 Pawlak 教授于 1982 年提出,是一种处理不精确、不确定和不完全数据的有效数学工具,能对数据进行分析 and 推理来发现隐含的知识、揭示潜在的规律<sup>[8]</sup>。基于

收稿日期:2018-06-21 返修日期:2018-10-06 本文受国家自然科学基金(61572406)资助。

**方波**(1991—),男,硕士生,主要研究方向为数据挖掘、机器学习等, E-mail:fangbo19910204@163.com; **陈红梅**(1971—),女,博士,教授,主要研究方向为粒计算与粗糙集、智能信息处理等, E-mail:hmchen@swjtu.edu.cn(通信作者); **王生武**(1995—),男,硕士生,主要研究方向为云计算与智能技术。

粗糙集的特征选择,即属性约简,是粗糙集理论的核心内容之一。通过属性约简,决策系统中待处理问题数据的维度大大降低,算法效率得到了有效提高。研究表明,粗糙集属性约简在处理维度较低、数据量较小的数据集时性能优异,但对于海量的、高维复杂的数据处理能力还不足<sup>[9]</sup>。

果蝇优化算法(Fruit Fly Optimization Algorithm, FOA)是由潘文超教授于2011年提出的一种源于果蝇觅食行为的新群智能优化算法<sup>[10]</sup>。群智能优化算法自身通常具有一定的全局优化和隐含并行性的特点,较为适用于求解 NP-Hard 问题<sup>[11]</sup>。为了能有效处理高维复杂数据并提高学习算法性能,专家们对不同适应度函数的构造、搜索策略的优化改进等方面进行了研究,提出了大量基于粗糙集和群智能优化算法的特征选择方法。Wang 等<sup>[12]</sup>结合了属性依赖度和特征子集大小来构造适应度函数,提出了一种基于粗糙集和粒子群优化算法的特征选择方法,并证明了其相比遗传算法,具有操作更简单、受特征空间大小影响更小、收敛效率更好的优点。王璐等<sup>[13]</sup>提出了一种基于粗糙集和蚁群优化算法的特征选择方法,该算法以特征子集长度和属性依赖度构建信息素更新策略,在较小的数据集下具有较好的性能,但随着数据集的增大,该算法容易陷入局部最优,效率下降。Chen 等<sup>[14]</sup>则基于信息熵和互信息的特征属性值提取启发式信息,提出了一种基于蚁群优化算法的粗糙集特征选择方法,从特征核出发搜索得到有效的特征子集。Chen 等<sup>[15]</sup>还提出了一种基于粗糙集和鱼群优化算法的特征选择方法,并证明了在搜索最小特征子集时该算法有较好表现,具有较强的全局搜索能力和较高的收敛效率。周涛等<sup>[9]</sup>基于代数和信息熵两个角度定义属性相关度,综合考虑特征子集数目、属性重要度、属性依赖度、特征编码等因素来构造通用的适应度函数框架,提出了一种基于 Rough Set 的高维特征选择混合遗传算法,在解决高维复杂的多目标优化问题上有较好的表现,但实际求解时需操作和调整的参数较多。Bae 等<sup>[16]</sup>提出了一种基于粗糙集的动态群智能优化算法,该算法结合了基于两阶段离散化改进的粒子群算法和粗糙集属性约简的思想,并利用 K-均值算法的思想对连续变量进行分组和聚类,将连续问题离散化后进行求解。实验表明:该算法在较高维数据集上能有效搜索特征子集,且在解决连续型问题上有较好的表现。

FOA 算法相比其他群智能优化算法具有操作更简单、需要设置和调整的参数更少、收敛速度更快等优点,国内外将 FOA 算法应用到特征选择领域的成果正逐渐增多<sup>[17-20]</sup>,但这部分研究主要利用了 FOA 算法简洁和快速收敛的优点,相应的改进也是对具体问题的参数选择是否合理进行调整以及在收敛精度的控制方面进行简单的优化,没有涉及 FOA 搜索方式整体结构的改进。同时,结合粗糙集理论的 FOA 特征选择方法的研究仍较少,因此本文提出了一种基于粗糙集和果蝇优化算法的特征选择方法(Feature Selection Based on Rough Sets and Fruit Fly Optimization Algorithm, RSFOAFS)。该方法利用一种新的双策略进化果蝇优化算法进行特征子集的迭代寻优,将果蝇群体动态地划分为精英子群和普通子群,对于精英子群,引入混沌变量引导果蝇个体在其附近搜索食物,优化其局部搜索能力,以防止算法陷入局部最优;对于普通子

群,引入权重因子改进标准 FOA 的随机搜索方式,扩大全局搜索,尝试在特征空间搜索具有最少信息损失的最小特征子集,并结合粗糙集属性依赖度和属性重要性理论构造适应度函数,对所选特征子集进行评估。在 UCI 数据集上的分组实验结果表明:本文提出的特征选择方法能有效地选出具有最优分类测试精度的最小特征子集。

## 2 相关基础理论

本节对粗糙集理论<sup>[21-24]</sup>的相关基础知识和标准果蝇优化算法<sup>[10]</sup>的框架进行介绍。

### 2.1 粗糙集

**定义 1(决策信息系统)** 决策信息系统  $S$  可表示为一个四元组  $S=(U, A, V, f)$ , 其中:

- 1)  $U=\{x_1, x_2, \dots, x_{|U|}\}$  是非空有限的对象集,称为论域;
- 2)  $A=\{a_1, a_2, \dots, a_{|A|}\}$  是非空有限的属性集,  $A=C \cup D$ ,  $C \cap D=\emptyset$ , 其中  $C$  是条件属性集,  $D$  是决策属性集;
- 3)  $V=\bigcup_{a \in A} V_a$  是决策信息系统属性值域,  $V_a$  表示属性  $a$  的值域;
- 4)  $f:U \times A \rightarrow V$  是信息函数,即  $\forall x \in U, a \in A, f(x, a) \in V_a$ , 表示论域中对象和属性的笛卡儿积与值域之间的映射关系。

**定义 2(等价关系)** 给定决策信息系统  $S=(U, A, V, f)$ , 对于任意条件属性子集  $B \subseteq A$ , 论域  $U$  上的一个不可分辨关系定义为:

$$IND(B)=\{(x, y) \in U \times U \mid \forall b \in B, f(x, b)=f(y, b)\} \quad (1)$$

其中,  $f(x, b)$  和  $f(y, b)$  分别表示对象  $x$  和对象  $y$  在条件属性  $b$  上的属性值。该不可分辨关系即一个等价关系,通常记为  $R_B$ 。

等价关系  $R_B$  在论域  $U$  上形成了一个划分  $U/R_B$ , 即等价类集合  $U/R_B=\{[x]_B \mid x \in U\}$ 。其中  $[x]_B$  表示对象集中由等价关系  $R_B$  所确定的等价类,即  $[x]_B=\{y \in U \mid (x, y) \in R_B\}$ 。在不引起歧义的情况下,  $U/R_B$  可简写为  $U/B$ 。

**定义 3(近似集)** 给定决策信息系统  $S=(U, A, V, f)$ , 对于任意对象集合  $X \subseteq U$  和  $U$  上的一个等价关系  $B$ ,  $X$  关于  $B$  的下近似集和上近似集分别定义为:

$$\underline{B}(X)=\{x \in U \mid [x]_B \subseteq X\} \quad (2)$$

$$\overline{B}(X)=\{x \in U \mid [x]_B \cap X \neq \emptyset\} \quad (3)$$

$X$  的下近似集、上近似集将论域  $U$  划分为 3 个不相交的区域,即正域  $POS_B(X)$ 、边界域  $BND_B(X)$  和负域  $NEG_B(X)$ , 分别定义如下:

$$POS_B(X)=\underline{B}(X) \quad (4)$$

$$BND_B(X)=\overline{B}(X)-\underline{B}(X) \quad (5)$$

$$NEG_B(X)=U-\overline{B}(X) \quad (6)$$

**定义 4(属性依赖度)** 给定决策信息系统  $S=(U, A, V, f)$ ,  $P, Q \subseteq A$  为论域  $U$  上的两个等价关系簇, 则  $Q$  的  $P$  正域记为  $POS_P(Q)=\bigcup_{x \in U/Q} \underline{B}(X)$ 。  $Q$  对  $P$  的依赖度  $k$  可定义为:

$$k=\gamma_P(Q)=\frac{|POS_P(Q)|}{|U|} \quad (7)$$

**定义 5(属性约简)** 给定决策信息系统  $S=(U, A, V, f)$ , 其中  $A=C \cup D$ , 设  $B \subseteq C$  为条件属性的任一子集, 则  $B$  被称为  $C$  的一个约简, 如果它满足:

$$1) POS_B(D) = POS_C(D);$$

$$2) \forall b \in B, POS_{B-b}(D) \neq POS_B(D).$$

以上关于粗糙集的理论都是基于集合运算定义的, 通常称为粗糙集理论的代数观点; 而从信息论的角度对粗糙集进行研究, 又有以下粗糙集理论的信息论观点<sup>[24-25]</sup>。

**定义 6** 给定决策信息系统  $S=(U, A, V, f)$ , 属性子集  $P, Q \subseteq A$  在论域  $U$  上导出的划分分别为  $X, Y (X = \{X_1, X_2, \dots, X_n\}, Y = \{Y_1, Y_2, \dots, Y_m\})$ , 则  $P, Q$  在  $U$  的子集组成的  $e$  代数上的概率分布为:

$$[X:p] = [X_1:p(X_1), X_2:p(X_2), \dots, X_n:p(X_n)] \quad (8)$$

$$[Y:p] = [Y_1:p(Y_1), Y_2:p(Y_2), \dots, Y_m:p(Y_m)] \quad (9)$$

其中,  $p(X_i) = \frac{|X_i|}{|U|}, i=1, 2, \dots, n; p(X_j) = \frac{|X_j|}{|U|}, j=1, 2, \dots, m$ 。

**定义 7(熵)** 知识(属性集合)  $P$  的熵  $H(P)$  定义为:

$$H(P) = - \sum_{i=1}^n p(X_i) \log(p(X_i)) \quad (10)$$

**定义 8(条件熵)** 知识(属性集合)  $Q(U|IND(Q)) = \{Y_1, Y_2, \dots, Y_m\}$  相对于知识(属性集合)  $P(U|IND(P)) = \{X_1, X_2, \dots, X_n\}$  的条件熵定义为:

$$H(Q|P) = - \sum_{i=1}^n p(X_i) \sum_{j=1}^m p(Y_j|X_i) \log(p(Y_j|X_i)) \quad (11)$$

其中,  $p(Y_j|X_i) = \frac{|Y_j \cap X_i|}{|X_i|}, i=1, 2, \dots, n, j=1, 2, \dots, m$ 。

**定义 9(属性重要性的信息论观点定义)** 给定决策信息系统  $S=(U, A, V, f)$ , 其中  $A=C \cup D$ , 任意条件属性子集  $B \subseteq C$ , 则对于任意条件属性  $r \in C - B$ , 其关于决策属性  $D$  的属性重要性  $SGF(r, B, D)$  定义为:

$$SGF(r, B, D) = H(D|B) - H(D|B \cup \{r\}) \quad (12)$$

$SGF(r, B, D)$  的值越大, 说明在已知  $B$  的条件下, 属性  $r$  对于决策  $D$  越重要。

## 2.2 果蝇优化算法

果蝇优化算法(FOA)是一种由果蝇觅食行为演化而来的搜索式全局优化进化算法。果蝇拥有比其他物种更优秀的嗅觉和视觉特性, 果蝇群体通过嗅觉收集漂浮在空气中的各种气味, 然后往食物的方向靠近, 并利用视觉辨别食物和同伴的位置, 往食物味道浓度高的位置聚合, 再利用嗅觉各自沿随机方向飞出寻找食物的具体位置, 如此循环直到找到食物为止。

标准果蝇优化算法结构简单, 易于实现, 其基本步骤如算法 1 所示。

### 算法 1 FOA 算法

Step 1 初始化参数: 群体规模 Sizepop, 最大迭代次数 Maxgen, 随机初始化果蝇群体位置  $X\_axis, Y\_axis$ 。

Step 2 赋予果蝇个体利用嗅觉搜寻食物的随机方向与距离, RandomValue 为搜索距离:

$$X_i = X\_axis + \text{RandomValue} \quad (13)$$

$$Y_i = Y\_axis + \text{RandomValue} \quad (14)$$

Step 3 由于无法得知食物位置, 因此先估计果蝇个体与远点的距离, 再计算果蝇个体味道浓度判定值, 此值为距离的倒数:

$$\text{Dist}_i = \sqrt{X_i^2 + Y_i^2} \quad (15)$$

$$S_i = 1/\text{Dist}_i \quad (16)$$

Step 4 将味道浓度判定值  $S_i$  代入味道浓度判定函数(或称适应度函数 fitness function), 求出果蝇个体位置的味道浓度:

$$\text{Smell}_i = \text{function}(S_i) \quad (17)$$

Step 5 找出该果蝇群体中味道浓度最低或者最高的果蝇个体(最优个体):

$$[\text{bestSmell bestindex}] = \min/\max(\text{Smell}_i) \quad (18)$$

Step 6 记录并保留最佳味道浓度值 bestSmell 与其 X 坐标和 Y 坐标, 与此同时种群利用视觉向该位置飞去:

$$\text{Smellbest} = \text{bestSmell} \quad (19)$$

$$X\_axis = X(\text{bestindex}) \quad (20)$$

$$Y\_axis = Y(\text{bestindex}) \quad (21)$$

Step 7 进入迭代寻优, 重复执行步骤 Step 2 - Step 5, 并判断最佳味道浓度是否优于前一迭代的最佳味道浓度。若当前迭代次数小于最大迭代次数 Maxgen, 则执行 Step 6。

## 3 基于粗糙集和 DSEFOA 算法的特征选择方法

标准 FOA 算法具有框架简单、参数较少、操作简单等优点, 但也存在过早收敛、易陷入局部最优的缺点。尤其对于多目标优化任务, 由于在整个迭代过程中, 果蝇群体只向当前最优果蝇个体学习, 群体之间缺乏信息共享和交流, 因此 FOA 算法极易早熟收敛, 从而陷入局部最优。为解决该问题, 本文引入一种新的双策略进化果蝇优化算法(DSEFOA)作为特征子集选择框架, 并基于粗糙集属性依赖度和属性重要性构造适应度函数引导迭代寻优, 具体思想如下。

### 3.1 DSEFOA 算法思想

DSEFOA 算法提出了一种新的群体分割策略, 动态地将整个果蝇群体划分为味道浓度大小不同的两个子群并分别进行寻优, 从而避免所有其他果蝇个体只向最优个体学习导致陷入局部最优的问题。

**定义 10(群体分割策略)** 设第  $t$  次迭代的果蝇群体为  $POP(t)$ , 群体规模为  $Sizepop$ 。利用式(15)一式(17)计算得到果蝇个体味道浓度  $Smell_i$ , 并分别得到最佳果蝇味道浓度  $Smell_{best}$  和最差味道浓度  $Smell_{worst}$ 。令  $d_1 = |Smell_i - Smell_{best}|, d_2 = |Smell_i - Smell_{worst}|$  分别表示果蝇个体  $i$  与味道浓度最佳的个体和最差的个体之间的距离。若  $d_1$  和  $d_2$  满足  $d_1 \leq \exp(-\frac{Iter}{Iter_{max}}) \cdot d_2$  (其中  $Iter$  为当前迭代次数,  $Iter_{max}$  为最大迭代次数), 则果蝇  $i$  被划分到精英子群  $POPe(t)$ ; 否则, 果蝇  $i$  被划分到普通子群  $POP_o(t)$ , 依次将规模为  $Sizepop$  的果蝇群体完成划分。

**定义 11** 设第  $t$  次迭代的果蝇精英子群为  $POPe(t)$ , 果蝇个体  $i$  的位置表示为  $S_{i,t} = (X_{i,axis}, Y_{i,axis})$ , 则对于所有的  $i \in POPe(t)$ , 有:

$$S_{i,t+1} = S_{i,t} + \alpha \cdot (S_{i,t} - S_i^*) \quad (22)$$

其中,  $S_{i,t+1}$  表示第  $t+1$  次迭代果蝇个体  $i$  将处于的位置,  $\alpha = \text{chaos}()$  表示混沌变量参数,  $S_i^*$  表示当代味道浓度最优的果蝇个体位置。

**定义 12** 设第  $t$  次迭代的果蝇普通子群为  $POP_o(t)$ , 果蝇个体  $i$  的位置表示为  $S_{i,t} = (X_{i\_axis}, Y_{i\_axis})$ , 则对于所有的  $i \in POP_o(t)$ , 有:

$$S_{i,t+1} = \omega_1 \cdot S_{i,t} + 2 \cdot \omega_2 \cdot (RandomValue - 0.5) \quad (23)$$

其中,  $S_{i,t+1}$  表示第  $t+1$  次迭代果蝇个体  $i$  将处于的位置, 权重因子  $\omega_1$  用以衡量果蝇个体  $i$  的历史认知,  $\omega_2$  用以动态调整个体搜索步长。在迭代寻优的前期, 位于普通子群的果蝇个体寻优结果差, 则需要较大的  $\omega_1$  和  $\omega_2$  以使得果蝇个体的搜索范围变大, 以尽快找到最优解; 而在迭代后期, 需要较小的  $\omega_1$  和  $\omega_2$  值, 以使群体在局部范围精细搜索, 加快收敛。因此对  $\omega_1$  和  $\omega_2$  做如下设置:

$$\omega = \omega_{\max} - \frac{\omega_{\max} - \omega_{\min}}{Iter_{\max}} \cdot t \quad (24)$$

其中,  $\omega_1$  和  $\omega_2$  的最大值  $\omega_{\max}$  和最小值  $\omega_{\min}$  根据实验情况分别设置,  $Iter_{\max}$  表示最大迭代次数,  $t$  表示当前迭代数。

DSEFOA 算法首先利用定义 10 给定的群体分割策略分割果蝇群体, 然后针对不同子群, 分别利用定义 11 和定义 12 中的进化策略进行寻优, 子群之间通过味道浓度最优果蝇个体位置的更新和群体的重新分割完成信息的交换。

### 3.2 基于粗糙集理论的适应度函数

合适的适应度函数对特征选择方法具有重要的作用。本文首先采用的适应度函数<sup>[12]</sup>如下:

$$Fitness = \alpha \cdot \gamma_B(D) + \beta \cdot \frac{|C| - |B|}{|C|} \quad (25)$$

其中,  $\alpha + \beta = 1$  且  $\alpha, \beta \in [0, 1]$ ,  $|C|$  为数据集特征属性的总个数,  $|B|$  为当前所选特征子集长度,  $\gamma_B(D)$  为属性依赖度。该适应度函数由属性依赖度和特征子集长度构成, 以保证在特征选择过程中, 所选取的特征子集具有较好的属性依赖度, 同时具有较少的特征个数。但从实验中可以发现, 属性依赖度对于特征选择更为重要, 根据经验, 通常设置  $\alpha = 0.9, \beta = 0.1$ 。算法的目的是最大化该适应度函数  $Fitness$ 。

该适应度函数已被广泛应用于各种基于粗糙集属性依赖度的特征选择方法研究中, 并取得了较好的成果。但进一步研究可以发现, 该函数还是存在一些理论上的不足。通过分析可知, DSEFOA 算法是在特征空间随机搜索可能的特征子集进行适应度的计算, 并选择适应度高且相对较少的特征子集作为最终的结果。显然适应度函数中的属性依赖度只针对已选取的特征计算, 而缺少对特征空间中其他特征属性的分析和评价; 另外, 从信息论的观点来看, 如果一个属性的增加不改变论域中本身已确定分类的实例, 且所有本身不能确定分类的实例仍然不能确定分类, 但是不确定性有所变化, 那么该属性的重要性在代数定义下为 0, 而在信息定义下不为 0<sup>[25]</sup>。因此本文引入基于信息熵的属性重要性, 结合属性依赖度构造第 2 个适应度函数:

$$Fitness' = \lambda_1 \cdot \gamma_B(D) + \lambda_2 \cdot \frac{1}{1 + e^{SGF(C-B)}} + \lambda_3 \cdot \frac{|C| - |B|}{|C|} \quad (26)$$

其中,  $\lambda_1, \lambda_2, \lambda_3$  为权重因子,  $\lambda_1 + \lambda_2 + \lambda_3 = 1$  且  $\lambda_1, \lambda_2, \lambda_3 \in [0, 1]$ ;  $\gamma_B(D)$  为属性依赖度;  $SGF(C-B)$  为所选特征子集  $B$  之外的条件特征集合的属性重要性;  $|C|$  为条件特征属性的总

个数;  $|B|$  为所选择的特征子集的特征数目。

该适应度函数不仅利用粗糙集属性依赖度对已选择的特征子集进行评价, 同时基于属性重要性对特征空间内所选特征之外的特征进行了评价, 依此选出具有最大依赖度、余下特征属性重要性最小以及最小长度的特征子集。实验中设置  $\lambda_1 = 0.7, \lambda_2 = 0.2, \lambda_3 = 0.1$ , 该算法的目的是最大化该适应度函数  $Fitness'$ 。

### 3.3 基于粗糙集和 DSEFOA 算法的特征选择方法

本文利用 DSEFOA 算法搜索特征子集, 引入粗糙集属性依赖度(以及基于信息观点的属性重要性)来构造适应度函数, 并将其作为启发式信息引导算法进行迭代寻优。特征子集选择的主要过程可分为以下几个步骤:

- 1) 随机产生  $N$  个 0-1 向量作为初始果蝇群体;
- 2) 利用 3.2 节给出的适应度函数计算果蝇的适应度值;
- 3) 根据适应度值, 比较并更新最优果蝇个体的位置;
- 4) 利用 DSEFOA 算法改进的搜索方法(定义 10、式(22)式(23))进行迭代搜索;

5) 若达到最大迭代次数或设定的阈值, 则终止算法并输出当前结果, 否则转入步骤 2)。

以上过程描述了本文过滤式特征选择方法的特征子集搜索过程和评价准则应用的框架, 即利用 DSEFOA 算法和本文给出的适应度函数搜索并评价最有效的特征子集。

基于粗糙集和 DSEFOA 算法的特征选择方法的详细描述如算法 2 所示。

#### 算法 2 基于粗糙集和 DSEFOA 算法的特征选择方法

输入: 决策信息系统  $S = (U, C \cup D, V, f)$ , 果蝇种群大小  $Sizepop$ , 最大迭代次数  $Iter_{\max}$ , 混沌因子  $\alpha$ , 权重因子  $\omega_1, \omega_2$

输出: 特征子集  $C' \subseteq C$

1. 随机初始化大小为  $N$  的果蝇种群  $POP = \{p_1, p_2, \dots, p_N\}$  及其初始位置  $S = \{S_1, S_2, \dots, S_N\}$
2.  $Iter \leftarrow 0$
3. while  $Iter < Iter_{\max}$  and  $\exists p_i \in POP, Smell_i > Smell_{best}$  do
4. for each  $p_i \in POP$  do
5. compute  $\gamma_{p_i}(D)$  and  $Smell_i \leftarrow \lambda_1 \cdot \gamma_{p_i}(D) + \lambda_2 \cdot \frac{1}{1 + e^{\frac{SGF(C-p_i)}}} + \lambda_3 \cdot \frac{|C| - |p_i|}{|C|}$
6. if  $Smell_i > Smell_{best}$  then
7.  $Smell_{best} \leftarrow Smell_i$
8.  $S_{best} \leftarrow S_i$
9. end if
10. if  $Smell_i < Smell_{worst}$  then
11.  $Smell_{worst} \leftarrow Smell_i$
12. end if
13. compute  $d_1 \leftarrow |Smell_i - Smell_{best}|$  and  $d_2 \leftarrow |Smell_i - Smell_{worst}|$
14. if  $d_1 \leq \exp(-\frac{Iter}{Iter_{\max}}) \cdot d_2$  then
15.  $POPe(Iter) \leftarrow POPe(Iter) \cup p_i$
16. end if
17. if  $d_1 > \exp(-\frac{Iter}{Iter_{\max}}) \cdot d_2$  then
18.  $POPo(Iter) \leftarrow POPo(Iter) \cup p_i$
19. end if

```

20. end for
21. for each  $p_i \in \text{POPe}(\text{Iter})$  do
22.    $S_{p_i, \text{Iter}+1} \leftarrow S_{p_i, \text{Iter}} + \alpha \cdot (S_{p_i, \text{Iter}} - S_{p_i}^*)$ 
23. end for
24. for each  $p_i \in \text{POPo}(\text{Iter})$  do
25.    $S_{p_i, \text{Iter}+1} \leftarrow \omega_1 \cdot S_{p_i, \text{Iter}} + 2 \cdot \omega_2 \cdot (\text{RandValue} - 0.5)$ 
26. end for
27.  $\text{Iter} \leftarrow \text{Iter} + 1$ 
28. end while
29.  $C' \leftarrow S_{\text{best}}$ 
30. output  $C'$ 

```

实验将在算法第 5 步分别使用式(25)及式(26)进行两组实验。

#### 4 实验及结果分析

为验证本文提出的特征选择方法的有效性,选取了规模大小各异、特征数目不同的 14 个 UCI 数据集进行实验,数据集详细信息如表 1 所列,该表给出了所选 UCI 数据集的样本数、特征个数以及类别数。

表 1 实验所用的 UCI 数据集的基本信息

Table 1 Basic information of UCI datasets used in experiments

ID	Dataset	Instances	Features	Class
1	arrhythmia	452	279	16
2	car	1728	6	4
3	chess	3196	36	2
4	choice	1473	9	3
5	credit approval	690	15	2
6	ionosphere	351	34	2
7	mushroom	8124	22	2
8	nursery	12960	8	5
9	plant leaves	1600	64	100
10	polish	10503	64	5
11	sonar	208	60	2
12	soybean(large)	307	35	19
13	voting	435	16	2
14	wine	178	13	3

表 3 特征子集分类精度的比较

Table 3 Comparison of classification accuracy of feature subsets

ID	Dataset	RSFOAFS		FOA		CFOA		PSO		RS-FS	
		Best	Avg	Best	Avg	Best	Avg	Best	Avg	Best	Avg
1	arrhythmia	0.6289	0.6090	0.6142	0.5653	0.6192	0.5744	<b>0.6382</b>	<b>0.6213</b>	0.5123	0.4986
2	car	<b>0.8968</b>	<b>0.8874</b>	0.8799	0.8772	0.8799	0.8787	0.8799	0.8787	0.8799	0.8783
3	chess	<b>0.9865</b>	<b>0.9726</b>	0.9456	0.9445	0.9566	0.9456	0.9651	0.9516	0.9566	0.9452
4	choice	<b>0.5447</b>	<b>0.5361</b>	0.5316	0.5222	0.5366	0.5268	0.5394	0.5311	—	—
5	credit approval	<b>0.8625</b>	<b>0.8611</b>	0.8551	0.8482	0.8597	0.8526	0.8595	0.8522	0.8585	0.8545
6	ionosphere	<b>0.9359</b>	<b>0.9288</b>	0.9117	0.9045	0.9137	0.9098	0.9301	0.9214	0.9356	0.9216
7	mushroom	<b>1.0000</b>	<b>0.9999</b>	0.9964	0.9937	1.0000	0.9999	1.0000	0.9999	1.0000	0.9948
8	nursery	<b>0.9710</b>	<b>0.9616</b>	0.9710	0.8959	0.9710	0.9596	0.9710	0.9587	0.6981	0.6954
9	plant leaves	<b>0.4297</b>	<b>0.4275</b>	0.4132	0.4045	0.4246	0.4201	0.4258	0.4214	0.2177	0.2201
10	polish	0.9521	0.9499	0.9376	0.9329	0.9501	0.9446	<b>0.9605</b>	<b>0.9528</b>	0.9301	0.9239
11	sonar	0.8046	0.7563	0.7453	0.7237	0.7598	0.7374	<b>0.8084</b>	<b>0.7689</b>	0.8062	0.7587
12	soybean(large)	<b>0.9910</b>	<b>0.9867</b>	0.9832	0.9711	0.9838	0.9755	0.9846	0.9778	0.9845	0.9769
13	voting	0.9416	<b>0.9369</b>	0.9401	0.9312	0.9406	0.9327	<b>0.9420</b>	0.9335	0.9371	0.9308
14	wine	<b>0.9588</b>	<b>0.9413</b>	0.9522	0.9213	0.9522	0.9287	0.9578	0.9392	0.9573	0.9407

从表 2 和表 3 可以看出,本文提出的特征选择方法除了在特征总数达 279 的 arrhythmia 数据集上的表现稍劣于 PSO 算法外,在大多数数据集(mushroom, credit approval, wine, choice, chess, car, soybean 和 voting)上都可选出最小的特征

基于表 1 中的 14 个常用 UCI 数据集,本文分别使用适应度函数式(25)、式(26)进行两组不同的实验,以验证本文提出的基于粗糙集和 DSEFOA 算法的特征选择方法的有效性。为了降低果蝇初始随机位置对最终选出的特征子集的影响,实验设置果蝇群体规模为特征总数的一半(数据集 arrhythmia 特征数目过大,因此设置为 50),迭代次数为 100 次,此外每个适应度函数都将进行 5 次独立实验,取平均值作为实验的最终数据。同时,本文使用的分类算法为 weka<sup>[26]</sup> 中的 J48 算法,采用十折交叉验证方法测定分类精度。实验选择的对比算法为标准 FOA 算法<sup>[10]</sup>、CFOA 算法<sup>[27]</sup>、PSO 算法<sup>[12]</sup> 以及经典粗糙集算法 RS-FS<sup>[15]</sup>。实验将对各算法所选特征子集的大小和分类精度。

#### 4.1 第 1 组实验

该组实验选取的适应度函数为:  $Fitness = \alpha \cdot \gamma_B(D) + \beta \cdot \frac{|C| - |B|}{|C|}$ , 同时设置  $\alpha = 0.9, \beta = 0.1$ 。表 2 和表 3 分别给出了不同算法所选出的特征子集的大小和特征子集的分类精度。

表 2 特征子集大小的比较

Table 2 Comparison of number of feature subsets

ID	Dataset	RSFOAFS		FOA		CFOA		PSO		RS-FS	
		Min	Avg	Min	Avg	Min	Avg	Min	Avg	Min	Avg
1	arrhythmia	52	59	69	74	67	69	48	56	<b>24</b>	<b>26</b>
2	car	<b>3</b>	<b>4</b>	5	5	5	5	5	5	5	5
3	chess	<b>26</b>	<b>27</b>	30	31	29	30	29	29	29	30
4	choice	<b>6</b>	<b>7</b>	7	8	7	8	7	7	—	—
5	credit approval	<b>8</b>	<b>9</b>	10	11	9	10	9	11	10	11
6	ionosphere	8	10	18	18	15	16	<b>7</b>	10	8	<b>9</b>
7	mushroom	<b>4</b>	<b>4</b>	4	5	4	5	4	4	4	5
8	nursery	7	7	1	7	7	8	7	7	<b>1</b>	<b>1</b>
9	plant leaves	13	16	34	37	20	26	16	19	<b>6</b>	<b>7</b>
10	polish	7	8	14	17	11	15	8	9	<b>4</b>	<b>4</b>
11	sonar	<b>6</b>	<b>8</b>	21	23	12	14	6	8	6	<b>7</b>
12	soybean (large)	<b>10</b>	<b>10</b>	12	14	11	13	10	11	12	13
13	voting	7	<b>8</b>	9	11	9	10	9	10	8	9
14	wine	<b>5</b>	<b>6</b>	8	10	8	10	7	9	6	6

子集,同时得到最优的分类精度,尤其在特征总数较高并且类别高达 100 次的 plant leaves 数据集上本文方法表现出了整体的优势;同时我们发现经典粗糙集算法 RS-FS 虽然在特征数目较高的 arrhythmia, sonar, ionosphere, plant leaves 和

polish 数据集上能得到相对最小的数据集,但算法的整体分类精度偏低,且在 choice 数据集上无法选出特征子集,这是由于在启发式算法中每次迭代时根据单个属性计算属性依赖度,当所有属性的属性依赖度都相近并低于阈值时,则在迭代过程中无法判断加入合适的属性。从该组实验可以看到,本文特征选择方法在较高特征数目和复杂分类问题上具有良好的性能。

为了更好地说明本文特征选择方法性能的普遍优越性,下面给出详细的 Friedman 检验和 Nemenyi 后续检验结果(本文选用显著性水平)。

首先,根据表 3 的分类精度,对实验涉及的 5 个算法在 14 个数据集上测得的分类精度进行排序并赋值(若算法测试性能相同则评分序值)。表 4 给出了每个算法在每个数据集上的测试性能的排列序值以及平均序值。

表 4 不同数据集上算法性能的排序结果

Table 4 Sorting results of performance of algorithms on different datasets

ID	Dataset	RSFOAFS	FOA	CFOA	PSO	RS-FS
1	arrhythmia	2	3	4	1	5
2	car	1	5	2.5	2.5	4
3	chess	1	5	3	2	4
4	choice	1	4	3	2	5
5	credit approval	1	5	3	4	2
6	ionosphere	1	5	4	3	2
7	mushroom	2	5	2	2	4
8	nursery	1	4	2	3	5
9	plant leaves	1	4	3	2	5
10	polish	2	4	3	1	5
11	sonar	3	5	4	1	2
12	soybean(large)	1	5	4	2	3
13	voting	1	4	3	2	5
14	wine	1	5	4	3	2
average value		1.3571	4.5000	3.1786	2.1786	3.7857

若算法性能相同,则其平均序值应该相同,且第  $i$  个算法的平均序值  $r_i$  服从自由度为  $k-1$  和  $(k-1)(N-1)$  的  $F$  分布

$$\tau_F = \frac{(N-1)\tau_x^2}{N(k-1) - \tau_x^2}$$

根据 Friedman 检验可知,“所有算法的性能相同”这个假设被拒绝,各类算法性能显著不同,则后续须进行 Nemenyi 检验。

利用 Nymenyi 检验计算平均序值差别的临界值域  $CD$ ,

$$CD = qa \sqrt{\frac{k(k+1)}{6N}}$$

其中  $k$  为算法个数,  $N$  为数据集个数。

在查表后得算法个数  $k=5$  的  $qa$  值为 2.4590,于是得到临界值  $CD=1.4695$ 。比较本算法与其他几类算法两两之间的平均序值的差值(与 FOA 算法、CFOA 算法、PSO 算法以及 RS-FS 算法的差值分别为 3.1429, 1.8251, 0.8215 和 2.4286)可知,除与 PSO 算法的平均序值小于  $CD$  值外,其他所有差值皆大于  $CD$  值,“两个算法性能相同”的假设被拒绝,说明本文特征选择方法的性能显著优于其他算法,但与 PSO 算法相比优势并不明显。为了体现本文方法在多目标分类问题上的性能,基于适应度函数式(26)进行第 2 组实验。

#### 4.2 第 2 组实验

该组实验选取的适应度函数为:  $Fitness' = \omega_1 \cdot \gamma_B(D) +$

$$\omega_2 \cdot \frac{1}{1 + e^{SGF(C-B)}} + \omega_3 \cdot \frac{|C| - |B|}{|C|}$$

同时设置  $\omega_1 = 0.7, \omega_2 =$

$0.2, \omega_3 = 0.1$ 。表 5 和表 6 分别给出了不同算法所选出的特征子集大小和特征子集分类精度。

表 5 特征子集大小的比较

Table 5 Comparison of number of feature subsets

ID	Dataset	RSFOAFS		FOA		CFOA		PSO		RS-FS	
		Min	Avg	Min	Avg	Min	Avg	Min	Avg	Min	Avg
1	arrhythmia	46	52	54	61	51	57	44	50	<b>24</b>	<b>26</b>
2	car	<b>3</b>	<b>4</b>	5	5	5	5	5	5	5	5
3	chess	<b>21</b>	<b>24</b>	29	30	27	28	27	28	29	29
4	choice	<b>6</b>	<b>7</b>	7	9	7	8	7	7	—	—
5	credit approval	<b>8</b>	<b>9</b>	10	11	9	10	9	11	10	11
6	ionosphere	<b>6</b>	<b>8</b>	15	16	13	15	7	9	7	9
7	mushroom	<b>4</b>	<b>4</b>	4	5	4	5	4	4	4	5
8	nursery	7	7	1	7	7	7	7	7	<b>1</b>	<b>1</b>
9	plant leaves	10	14	29	34	11	21	13	16	<b>6</b>	<b>7</b>
10	polish	5	6	13	16	10	12	6	8	<b>4</b>	<b>4</b>
11	sonar	<b>6</b>	<b>7</b>	16	18	11	12	6	7	6	7
12	soybean (large)	<b>9</b>	<b>10</b>	11	14	10	13	9	10	11	12
13	voting	7	<b>8</b>	9	11	9	10	9	10	8	9
14	wine	<b>5</b>	<b>6</b>	8	10	8	10	7	9	6	6

表 6 特征子集分类精度的比较

Table 6 Comparison of classification accuracy of feature subsets

ID	Dataset	RSFOAFS		FOA		CFOA		PSO		RS-FS	
		Best	Avg	Best	Avg	Best	Avg	Best	Avg	Best	Avg
1	arrhythmia	0.6452	0.6355	0.6214	0.5988	0.6377	0.6212	0.6478	0.6391	0.5123	0.5026
2	car	<b>0.8968</b>	<b>0.8874</b>	0.8799	0.8772	0.8799	0.8787	0.8799	0.8787	0.8799	0.8791
3	chess	<b>0.9865</b>	<b>0.9726</b>	0.9456	0.9445	0.9566	0.9456	0.9651	0.9516	0.9566	0.9452
4	choice	<b>0.5447</b>	<b>0.5361</b>	0.5316	0.5222	0.5366	0.5268	0.5394	0.5311	—	—
5	credit approval	<b>0.8625</b>	<b>0.8611</b>	0.8551	0.8482	0.8597	0.8526	0.8595	0.8522	0.8585	0.8545
6	ionosphere	<b>0.9359</b>	<b>0.9288</b>	0.9117	0.9045	0.9137	0.9098	0.9301	0.9215	0.9356	0.9216
7	mushroom	<b>1.0000</b>	<b>1.0000</b>	0.9964	0.9937	1.0000	0.9999	1.0000	0.9999	1.0000	0.9948
8	nursery	<b>0.9710</b>	<b>0.9623</b>	0.9710	0.8975	0.9710	0.9596	0.9710	0.9587	0.6981	0.6954
9	plant leaves	<b>0.4457</b>	<b>0.4362</b>	0.4132	0.4045	0.4266	0.4221	0.4246	0.4219	0.2177	0.2201
10	polish	0.9502	<b>0.9483</b>	0.9376	0.9329	0.9501	0.9446	<b>0.9527</b>	0.9478	0.9301	0.9239
11	sonar	0.8092	0.7824	0.7562	0.7317	0.7652	0.7398	0.8084	0.7715	0.8100	0.7587
12	soybean(large)	<b>0.9910</b>	<b>0.9879</b>	0.9840	0.9735	0.9854	0.9772	0.9862	0.9788	0.9833	0.9761
13	voting	0.9416	<b>0.9372</b>	0.9401	0.9312	0.9416	0.9337	<b>0.9420</b>	0.9335	0.9371	0.9308
14	wine	<b>0.9588</b>	<b>0.9418</b>	0.9522	0.9213	0.9568	0.9327	0.9524	0.9292	0.9573	0.9407

从表5和表6可以看出,除特征总数较高的 arrhythmia, plant leaves, polish 和 nursery 数据集外,本文特征选择方法均能选出相对最小的特征子集,且除 arrhythmia 数据集外,选出的特征子集均有最好的分类精度。在调整适应度函数之后,本文算法综合考虑了已选取特征属性的属性依赖度和剩下未选取特征属性的重要性,不仅突出了所选特征子集的重要性,还考虑了整个特征空间中其余特征的有效程度,配合衡量特征子集大小的度量,很好地体现了所选特征子集对特征空间的区分度。对比第1组实验的结果可以发现,本文特征选择方法不仅在较低特征数目的数据集上选出了更小的特征子集,得到了更好的分类精度,还在维数较高的数据集(sonar, plant leaves 以及 polish)上表现出了更好的性能,仅仅在特征数目高达279的 arrhythmia 数据集上的表现稍劣于 PSO 算法,但整体优于其他算法。

同样地,为了进一步说明本文特征选择方法的普遍优越性,给出详细的 Friedman 检验和 Nemenyi 后续检验结果。

算法性能排序结果如表7所列,其中给出了每个算法在每个数据集上的测试性能的排列序值以及平均序值。

比较本算法与其他几类算法两两之间的平均序值的差值(与 FOA 算法、CFOA 算法、PSO 算法以及 RS-FS 算法的差值分别为 3.4973,1.8545,1.5688 和 2.7116)可知,其差值均大于临界值域  $CD=1.4695$ ,这充分说明本文特征选择方法的性能相比其他算法具有显著优越性。

表7 不同数据集上算法性能的排序结果

Table 7 Sorting results of performance of algorithms on different datasets

ID	Dataset	RSFOAFS	FOA	CFOA	PSO	RS-FS
1	arrhythmia	2	4	3	1	5
2	car	1	5	2.5	2.5	4
3	chess	1	5	3	2	4
4	choice	1	4	3	2	5
5	credit approval	1	5	3	4	2
6	ionosphere	1	5	4	3	2
7	mushroom	1	5	2.5	2.5	4
8	nursery	1	4	2	3	5
9	plant leaves	1	4	2	3	5
10	polish	1	4	3	2	5
11	sonar	1	5	4	2	3
12	soybean(large)	1	5	4	3	2
13	voting	1	4	2	3	5
14	wine	1	5	3	4	2
average value		1.0741	4.5714	2.9286	2.6429	3.7857

通过分析以上两组实验结果(表2—表7)可以看到,本文提出的特征选择方法可以解决经典粗糙集 RS-FS 属性约简算法对部分数据集无法选出有效特征子集或所选的特征子集不具有有效分类精度的问题;同时,通过分类精度测试的对比结果可以发现,本文算法相比 FOA 算法、CFOA 算法和 PSO 算法在选取最小特征子集和得到最优分类精度上更具优越性。在上述大部分情况下,本文特征选择方法都选出了相对更好的特征子集并得到了更优的分类精度,这说明了本文提出的特征选择方法的有效性。

**结束语** 本文基于粗糙集属性依赖度和属性重要性理论,提出了一种基于粗糙集和果蝇优化算法的特征选择方法

(RSFOAFS)。该方法使用二进制编码特征空间,利用改进的果蝇优化算法(DSEFOA)进行特征子集的迭代寻优,并结合粗糙集属性依赖度和属性重要性构造适应度函数,对所选特征子集进行评估,Friedman 检验和 Nemenyi 后续检验的结果证明了本文方法所选取的有效特征子集具有良好的性能,可得到较高的分类测试精度。

同时,在研究过程中可以发现,基于群智能优化算法的特征选择方法具有一定的内在并行性,在综合考虑实验操作复杂度和时间开销后,下一步工作将集中于特征选择方法的并行化研究。

## 参考文献

- [1] LI J D, LIU H. Challenges of feature selection for big data analytics[J]. IEEE Intelligent Systems, 2017, 32(2): 9-15.
- [2] MIAO J Y, NIU L F. A survey on feature selection[J]. Procedia Computer Science, 2016, 91: 919-926.
- [3] CHANDRASHEKAR G, SAHIN F. A survey on feature selection methods[J]. Computers and Electrical Engineering, 2014, 40(1): 16-28.
- [4] LI M, KAMILI M. Research on feature selection methods and algorithms[J]. Computer Technology and Development, 2013(12): 16-21. (in Chinese)  
李敏, 卡米力·木依丁. 特征选择方法与算法的研究[J]. 计算机技术与发展, 2013(12): 16-21.
- [5] SANTANA L E A D S, CANUTO A M D P. Filter-based optimization techniques for selection of feature subsets in ensemble systems[J]. Expert Systems with Applications, 2014, 41(4): 1622-1631.
- [6] YANG P Y, WEI L, ZHOU B B, et al. Ensemble-based wrapper methods for feature selection and class imbalance learning[C]// Pacific-Asia Conference on Knowledge Discovery and Data Mining. Berlin, Heidelberg: Springer, 2013: 544-555.
- [7] YOU M Y, LIU J M, LI G Z, et al. Embedded feature selection for multi-label classification of music emotions[J]. International Journal of Computational Intelligence Systems, 2012, 5(4): 668-678.
- [8] PAWLAK Z, GRZYMALA-BUSSE J, SLOWINSKI R, et al. Rough sets[J]. International Journal of Computer and Information Science, 1982, 11(5): 341-356.
- [9] ZHOU T, LU H L, ZHANG Y N, et al. A new hybrid genetic algorithm for high dimension feature selection based on rough set[J]. Journal of Nanjing University(Natural Sciences), 2015, 51(4): 880-893. (in Chinese)  
周涛, 陆惠玲, 张艳宁, 等. 基于 Rough Set 的高维特征选择混合遗传算法研究[J]. 南京大学学报(自然科学), 2015, 51(4): 880-893.
- [10] PAN W T. A new fruit fly optimization algorithm: taking the financial distress model as an example[J]. Knowledge-Based Systems, 2012, 26(2): 69-74.
- [11] GAO H C, FENG B Q, ZHU L. Reviews of the meta-heuristic algorithms for TSP[J]. Control and Decision, 2006, 21(3): 241-247. (in Chinese)

- 高海昌,冯博琴,朱利. 智能优化算法求解 TSP 问题[J]. 控制与决策,2006,21(3):241-247.
- [12] WANG X Y, YANG J, TENG X L, et al. Feature selection based on rough sets and particle swarm optimization[J]. Pattern Recognition Letters,2007,28(4):459-471.
- [13] WANG L, QIU T R, HE N, et al. A method for feature selection based on rough sets and ant colony optimization algorithm[J]. Journal of Nanjing University(Natural Sciences),2010,46(5):487-493. (in Chinese)  
王璐,邱桃荣,何姐,等. 基于粗糙集和蚁群优化算法的特征选择方法[J]. 南京大学学报(自然科学),2010,46(5):487-493.
- [14] CHEN Y M, MIAO D Q, WANG R Z. A rough set approach to feature selection based on ant colony optimization[J]. Pattern Recognition Letters,2010,31(3):226-233.
- [15] CHEN Y M, ZHU Q X, XU H R. Finding rough set reducts with fish swarm algorithm [J]. Knowledge-Based Systems, 2015,81(C):22-29.
- [16] BAE C, YEH W C, CHUNG Y Y, et al. Feature selection with intelligent dynamic swarm and rough set [J]. Expert Systems with Applications,2010,37(10):7026-7032.
- [17] YUAN M, WANG M, PAN Y X, et al. A feature selection method for the milling force signal based on the improved fruit fly optimization algorithm[J]. Journal of Vibration and Shock, 2016,35(24):196-200. (in Chinese)  
袁敏,王玫,潘玉霞,等. 基于改进果蝇优化算法的铣削力信号特征选择方法[J]. 振动与冲击,2016,35(24):196-200.
- [18] YIN L J, LI X Y, GAO L, et al. A new improved fruit fly optimization algorithm for traveling salesman problem[C]//The 8th International Conference on Advanced Computational Intelligence. Chiang Mai, Thailand: IEEE,2016:21-28.
- [19] MENG T, PAN Q K. An improved fruit fly optimization algorithm for solving the multidimensional knapsack problem[J]. Applied Soft Computing,2017(50):79-93.
- [20] ZHANG Y W, CUI G M, WANG Y, et al. An optimization algorithm for service composition based on an improved FOA[J]. Tsinghua Science and Technology,2015,20(1):90-99.
- [21] ZHANG W X, WU W Z, LIANG J Y, et al. Rough set theory and method[M]. Beijing: Science Press, 2001:15-90. (in Chinese)  
张文修,吴伟志,梁吉业,等. 粗糙集理论与方法[M]. 北京:科学出版社,2001:15-90.
- [22] PAWLAK Z. Rough sets: theoretical aspects of reasoning about data; Vol 9[M]. Kluwer Academic Publishers,1992.
- [23] GUAN Y Y, WANG H K. Set-valued information systems[J]. Information Sciences,2006,176(17):2507-2525.
- [24] WANG G Y, YU H, YANG D C. Decision table reduction based on conditional information entropy[J]. Chinese Journal of Computers,2002,25(7):759-766. (in Chinese)  
王国胤,于洪,杨大春. 基于条件信息熵的决策表约简[J]. 计算机学报,2002,25(7):759-766.
- [25] LIANG J Y, WANG F, DANG C Y, et al. A group incremental approach to feature selection applying rough set technique[J]. IEEE Transactions on Knowledge and Data Engineering,2014,26(2):294-308.
- [26] WAIKATO M L G. Weka 3: Data Mining Software in Java[EB/OL]. <http://www.cs.waikato.ac.nz/ml/weka/>.
- [27] MITIC M, VUKOVIC N, PETROVIC M, et al. Chaotic fruit fly optimization algorithm [J]. Knowledge-Based Systems, 2015,89(C):446-458.