ITC 4/48
Information Technology
and Control
Vol. 48 / No. 4 / 2019
pp. 590-601
DOI 10.5755/j01.itc.48.4.22330

An Adaptive Hybrid Ant Colony Optimization Algorithm for the Classification Problem

Received 2018/12/20 | Accepted after revision 2019/10/09

http://dx.doi.org/10.5755/j01.itc.48.4.22330

An Adaptive Hybrid Ant Colony Optimization Algorithm for the Classification Problem

Anxiang Ma, Xiaohong Zhang, Changsheng Zhang, Bin Zhang
College of Computer Science & Engineering, Northeastern University,
Heping District, Wenhua Road No.11, Shenyang, China
E-mails: maanxiang@mail.neu.edu.cn, zhangxiaohong@mail.neu.edu.cn,
zhangchangsheng@mail.neu.edu.cn, zhangbin@mail.neu.edu.cn

Corresponding author: maanxiang@mail.neu.edu.cn, zhangchangsheng@mail.neu.edu.cn

Classification is an important data analysis and data mining technique. Taking into account the comprehensibility of the classifier generated, an adaptive hybrid ant colony optimization algorithm called A_HACO is proposed which can effectively solve classification problem and get the comprehensible classification rules at the same time. The algorithm incorporates the artificial bee colony optimization strategy into the ant colony algorithm. The ant colony global optimization process is used to adaptively select the appropriate rule evaluation function for the data set given. Based on the classification rules obtained, the artificial bee colony optimization strategy is used to tackle the continuous attributes for further optimization of classification rules. This approach is evaluated experimentally using different standard real datasets, and compared with some proposed related classification algorithms. It shows that A_HACO can adaptively select the appropriate rule evaluation function and has better accuracy compared with related works.

KEYWORDS: ant colony optimization, artificial bee colony optimization, classification, classification rule, rule evaluation function.

1. Introduction

Classification is an important data analysis technique, which is widely used in the areas such as image processing, pattern recognition and management science. Depending on whether the classifier generated is comprehensible or not, existing algorithms can be divided into two types which are classification algorithms with incomprehensible classifier (CAIC) and comprehensible classifier (CACC), respectively.

Complex and incomprehensible mathematical functions are generated by CAIC such as support vector machine [1, 4, 15]. A set of classification rules are generated by CACC such as decision trees learning [2, 20, 24]. In many real-life applications, both accuracy and comprehensibility are important so that domain experts can understand and validate classifier. Hence, this paper will focus on CACC.

CACC includes parallel covering algorithms [2, 20] and sequential covering algorithms [14, 17]. Parallel covering algorithms generate all classification rules at one time, such as decision trees learning. It generates a decision tree from training examples by a greedy method, then, converts the decision tree into a set of classification rules. Decision trees learning is easier to converge to a local optimum solution because of no backtracking. Sequential covering algorithms generate a classification rule with highest accuracy every time, and get the ordered rules set in an incremental mode. Meta-heuristic methods can effectively tackle optimization problems, so sequential covering algorithms based on meta-heuristic methods [3, 23] are more appealing to researchers. AntMiner [17] is the first classification algorithm based on ant colony optimization, which defines the heuristic information and pheromone information related to classification problems. However, the AntMiner can only tackle classification problems whose attributes are categorical, namely discrete classification problems. To improve the performance of AntMiner, algorithms [10, 14, 19] modify and extend the AntMinter from some aspects, such as pheromone updating function, heuristic information definition and rule evaluation function. Reference [16] is the classification algorithm based on genetic method, which defines selection, crossover and mutation operation according to the characteristics of classification problems. It can only tackle categorical attributes. PSO/ACO [5] and PSO/ACO2 [6] cooperate particle swarm optimization process with ant colony optimization process, which can tackle two kinds of attributes. In [21], the classification algorithm based on artificial bee colony can effectively tackle continuous attributes, but cannot tackle categorical attributes. Since the real-life classification problems usually include both categorical and continuous attributes, it is very meaningful for the research of classification algorithms which can tackle both kinds of attributes.

Moreover, the sequential covering algorithms use rule evaluation functions to evaluate the quality of rules. References [11-13] summarize some common rule evaluation functions, such as Klosgen measure, F-measure, Relative cost measure and M-estimate. Existing sequential covering algorithms based on meta-heuristic method all fix a certain rule evaluation function. By analyzing the studies [11-13], we can see that given a data set, the performance of the algorithm varies with rule evaluation function used, and there is no rule evaluation function that performs well on all considered data sets.

It is hard for the existing approaches to automatically identify an appropriate rule evaluation function during optimizing process for a given data set, and it results in low performance. In addition, the existing approaches are hard to handle both continuous and categorical attributes. To tackle these problems, an adaptive hybrid ant colony optimization algorithm called A_HACO is proposed in this paper. It combines the adaptive ant colony optimization process with the artificial bee colony optimization process. In the ant colony global optimization process, classification rules are incrementally constructed, and more importantly, the appropriate rule evaluation function can be automatically selected from the candidate rule evaluation functions based on the information obtained during the optimizing process. To effectively tackle the continuous attributes, an artificial bee colony optimization is designed and incorporated into the ant colony optimization process.

This paper is organized as follows. In Section 2, we give the details of the adaptive hybrid ant colony optimization algorithm for classification problem including ant colony optimization model and searching strategies. The evaluations of this approach including its parameters tuning and comparative studies based on different standard real datasets are given in Section 3. Finally, Section 4 summarizes the contribution of this paper along with some future research directions.

2. The A_HACO Algorithm

The adaptive hybrid ant colony optimization (A_HACO) algorithm for classification problems incorporates the artificial bee colony optimization process into the ant colony optimization process, which can incrementally construct accurate and comprehensible classification rules. In A_HACO, four candidate rule evaluation functions are considered, namely, Klosgen measure, F-measure, M-estimate and Q+ [16]. In the ant colony optimization process, each rule evaluation function owns a group of artificial ants and a rule set. In the beginning, each rule evaluation function has same amount of ants and rule set is null. Rule set is gradually expanded and artificial ants are dynamically adjusted according to the quality of rule evaluation function after generating a classification rule. Rule sets are compared with each other to determine the quality of rule evaluation functions. Rule evaluation functions with high quality are allocated more ants, and the number of ants corresponding to rule evaluation functions with low quality is decreased. When the amount of ants equals to zero, corresponding rule evaluation function is abandoned. The artificial bee colony optimization process is used to tackle continuous attributes to further optimize the classification rule obtained in the adaptive ant colony global optimization process. The above process is repeated until only the best rule evaluation function is remained and all classification rules are generated.

2.1. ACO and ABC Approaches

2.1.1. Ant Colony Optimization(ACO)

Ant colony optimization (ACO) [22] is a meta-heuristic algorithm motivated by ants' behavior, and has some notable features, such as distributed computing, information positive feedback and heuristic searching. Ant colony algorithm has been widely applied to tackle discrete optimization problems [8, 18].

ACO is composed of three modules, which are search space representation, probabilistic state transition, and pheromone updating respectively. The main difference among ant colony algorithms is pheromone updating strategy. This paper is based on Max-Min ant system to realize classification tasks, so it takes Max-Min ant system as an example to introduce three modules of ACO:

1 Search space representation and pheromone initialization. ACO algorithm uses a graph model to represent optimization problem domain, and each path represents a feasible solution of problems. Because of the absence of priori knowledge about the path quality at the beginning, most ACO algorithms give the same pheromone value for each edge of graph. Max-Min ant system sets upper and lower bounds for pheromone value, and uses upper bounds as initial pheromone value.

2 Probabilistic states transition. Define a problem dependent heuristic value, ants choose next vertex according to pheromone value ($\tau_{ij}(t)$) and heuristic value (η_{ij}). For the ant at vertex i, the probability P_{ij} of going to vertex j is defined by (1)

$$P_{ij}^k = \begin{cases} \dfrac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum\limits_{n \in allowd_k} \tau_{in}^{\alpha}(t)\eta_{in}^{\beta}(t)} & j \in allowed_k \\ 0 & otherwise \end{cases}, \tag{1}$$

where α and β are weight parameters that indicate the relative importance of the pheromone and heuristic value. The parameter $allowed_k$ denotes the vertexes to be visited by the ant k. The ant k uses parameter $tabu_k$ to record the vertexes visited already.

3 Pheromones update. The ant records the quality of paths by depositing pheromone. The more pheromone is deposited to the path with higher quality, otherwise, the less pheromone is deposited to the path with lower quality. In Max-Min ant system, the pheromone on the best path is reinforced, and the pheromone on other paths is evaporated. Pheromone value cannot exceed the upper and lower bounds.

In ACO algorithm, construction graph representing search space, and the definitions of pheromone and heuristic value are all related to the specific problem.

2.1.2. Artificial Bee Colony Optimization(ABC)

Artificial Bee Colony (ABC) is one of the most recently defined algorithms [9], inspired by the intelligent forage behavior of honey bees. ABC algorithm can find good solutions for continuous optimization problems, which has the features of faster convergence and robustness.

In ABC algorithm, the colony of artificial bees consists of three groups of bees: employed bees, onlookers and scouts. A food source represents a feasible solution for the optimization problem. The nectar amount of a food source corresponds to the quality of the solution. For every food source, there is only one employed bee.

In other words, the number of employed bees is equal to the number of food sources. The food source cannot be improved through the predetermined number of trials "*limit*", then that food source is abandoned by its employed bee and then the employed bee becomes a scout. The scout is responsible to search a new food source. The onlooker chooses a food source to search according to the nectar amount of the food source, and as the nectar amount of the food source increases, the probability with the preferred source by an onlooker bee increases proportionally. The probability with the food source to be chosen by an onlooker is defined by formula (2).

$$p_i = \frac{fit_i}{\sum_{k=1}^{SN} fit_k}, \tag{2}$$

where fit_i denotes the nectar amount of the food source i, and parameter SN denotes the amount of food sources, namely, the number of employed bees.

The onlooker has a local search around the food source X_i chosen so as to find a neighbor food source. Different ABC algorithms have different strategies of local search. The following formula (3) is one of strategies of local search:

$$V_{ij} = X_{ij} + \Phi_{ij}(X_{ij} - X_{kj}), \tag{3}$$

where X_{ij} denotes the j-th element of food source X_i chosen by the onlooker, V_{ij} denotes the j-th element of the neighbor food source V_i of X_i, Φ_{ij} is a random number between "-1" and "1", X_{kj} denotes the j-th element of a random food source X_k(k<>j). The algorithm compares V_{ij} with X_{ij}, if V_{ij} has an equal or better nectar amount than X_{ij}, then X_{ij} is replaced by V_{ij} in the memory. Otherwise, X_{ij} is retained in the memory. In other words, a greedy selection mechanism is employed as the selection operation between the old and the neighbor food sources.

In ABC algorithm, the definitions of food source and strategy of local search are related to the specific problem.

2.2. Ant Colony Optimization Model for Classification Problems

According to the features of classification problems, the contents related to optimization problems in ACO are defined, and they include construction graph, pheromone information and heuristic information.

In the construction graph for classification problems, each path represents a classification rule of the form **if** *rule antecedent* **then** *rule consequent*, where the rule antecedent is a conjunction of terms. The terms are of the form *Attribute=value*. Because ACO algorithm can only tackle categorical attributes, continuous attributes need to be discretized before constructing graph model. Based on decision trees learning [20], the paper converts a continuous attribute into a categorical attribute with two values according to the discretization rule of maximum information gain. Suppose that there are n attributes $A_1, A_2, \dots, A_n$ with attribute A_i has p_i values, the construction graph is defined as G=(V, Γ), where vertex set $V=\{v_{start}, v_{stop}\} \cup \{v_{i,j} | i=0, 1, 2, \dots, n; j=0, 1, 2, \dots, p_i\}$, where v_{start} and v_{stop} are start vertex and stop vertex, vertex $v_{i,j}$ denotes the j-th value of A_i, vertex $v_{i,0}=any$ denotes any value of A_i, vertex $v_{0,j}$ denotes the j-th value of categories; edge set $\Gamma=\{<v_{i,j}, v_{i+1,k}>, <v_{start}, v_{1,k}>, <v_{n,j}, v_{stop}>\}$, where i=1,2,...,n and k=0,1,2,...,p_i。

Heuristic information is used to measure the classification ability of a vertex, which is defined as follows:

$$\eta_{v_{i,j}}(t) = \frac{|majorityclass(T_{ij})|}{|T_{ij}|}, \tag{4}$$

where T_{ij} is the set of uncovered training samples having the value of A_i equal to $v_{i,j}$, and $majorityclass(T_{ij})$ is the set of training samples with the most common category in T_{ij}.

The ants deposit a certain amount of pheromone to the paths according to the quality of solutions represented by paths. Only the pheromone on the best path is reinforced, and the pheromone on the other paths is evaporated. Pheromone updating rule is defined by (5).

$$\tau(v_{i,j}, v_{i+1,k}) = \begin{cases} \rho \bullet \tau(v_{i,j}, v_{i+1,k}) + \Delta\tau_{belrule}^{best} & if \quad \rho \bullet \tau(v_{i,j}, v_{i+1,k}) + \Delta\tau_{belrule}^{best} \in (\tau_{\min}, \tau_{\max}) \\ \tau_{\max} & if \quad \rho \bullet \tau(v_{i,j}, v_{i+1,k}) + \Delta\tau_{belrule}^{best} \geq \tau_{\max} \\ \tau_{\min} & if \quad \rho \bullet \tau(v_{i,j}, v_{i+1,k}) + \Delta\tau_{belrule}^{best} \leq \tau_{\min} \end{cases},$$
$$\Delta\tau_{belrule}^{best} = \begin{cases} Rf_i(belrule) & if \quad belrule = rule_{best} \\ 0 & otherwise \end{cases}, \tag{5}$$

where ρ denotes the evaporation factor of pheromone, $\Delta\tau_{belrule}^{best}$ denotes pheromone increment, pheromone of the path is reinforced only when the rule *belrule* represented by the path is the best rule, $Rf_i()$ is the i-th rule evaluation function.

2.2.1. Rule Generating

Because the classification rule set varies with rule evaluation functions, each rule evaluation function owns a group of artificial ants, a construction graph and a rule set. References [11-13] summarize some common rule evaluation functions, such as Klosgen measure, F-measure and M-estimate. We select four rule evaluation functions with better performance as candidate rule evaluation functions in A_HACO, which are Klosgen measure, F-measure, M-estimate and Q^+.

Rule evaluation function is usually specified as a function $f(p, n, P, N)$, where p and n refer to respectively the number of correctly and incorrectly covered examples, and also refer to as true positives and false positives. P is the total number of examples of the target class remaining in the training set, while N is the total number of examples belonging to other classes.

1 Klosgen measure

$$f_K(\omega) = \left(\frac{p+n}{P+N}\right)^{\omega} \cdot \left(\frac{p}{p+n} - \frac{P}{P+N}\right). \tag{6}$$

The Klosgen measure is defined by (6). It multiplicatively trades off full coverage and precision. The parameter ω controls the weight assigned to the coverage. At $\omega = 0$, the Klosgen measure equals precision. At $\omega = 1$, the Klosgen heuristic is equivalent to the weighted relative accuracy. In the limit $\omega \to \infty$, Klosgen measure equals full coverage. In general, $\omega < 1$ is the optimal region for separate-and-conquer rule induction.

2 F-measure

$$f_F(\gamma) = \frac{(\gamma^2+1) \cdot \frac{p}{p+n} \cdot \frac{p}{P}}{\gamma^2 \cdot \frac{p}{p+n} \cdot \frac{p}{P}}. \tag{7}$$

The F-measure is defined by (7). It is the weighed harmonic mean of precision and coverage. At $\gamma=0$, f_F equals precision. At $\gamma=1$, Precision and recall are weighed equally. In the limit $\gamma \to \infty$, f_F becomes recall.

3 M-estimate

$$f_m(m) = \frac{p + m \cdot \frac{P}{P+N}}{p+n+m}. \tag{8}$$

The m-estimate is defined by (8). It assumes an a priori coverage of m examples with a distribution equal to the class distribution. At $m=0$, a priori coverage of zero leads to precision.

4 Q^+

$$f_{Q^+} = \frac{p}{p+n} + \frac{p}{P+N}. \tag{9}$$

Given a rule evaluation function, rule generating module is used to find an optimal classification rule. Rule generating module firstly loads corresponding construction graph, the number of ants and training samples. Then ants construct classification rules one by one, and the quality of classification rules is evaluated according to rule evaluation function. The pheromone on the best path is reinforced, and the pheromone on the other paths is evaporated. The above process is repeated until convergence is satisfied. The pseudo code of rule generating module is described as following.

Algorithm 1. FINDRULE(RuleEvalFunc)

```
ConstructGraph;   //Loading construction graph G*
LoadAntCount;   //Loading the number of ants
LoadExamples   //Loading training samples
Initialize Pheromone;
Repeat
    for i = 1 to AntCount do
      FindAPath; //Construct a classification rule
    UpdatePheromone;    //Update pheromone
    UpdateGlobalBestPath; //Update the best path
    IterationCount = IterationCount + 1;
    If IterationCount > LimitIteration
       Then break;
Until converged;
GlobalRule=GlobalBestPath;
SaveGraph;            //Save graph G*
return GlobalRule
```

Convergence occurs when all the edges of one path have a pheromone level τ_{max} and all others edges have pheromone level τ_{min}. The classification rule represented by the path with τ_{max} is the optimal rule. In order to prevent the algorithm running time too long, set a variable *IterationCount* recording the number of iteration. When *IterationCount* reaches a user-defined threshold *LimitIteration*, the iteration is forcibly ended, and the classification rule represented by *GlobalBestPath* is considered as the optimal rule.

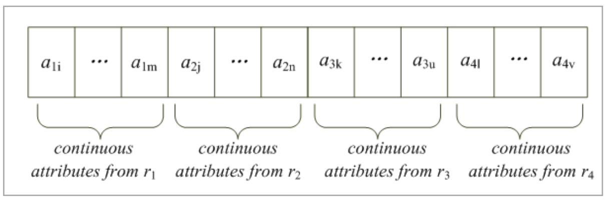
2.2.2. Artificial Bee Colony Optimization Process

To further optimize the classification rule generated, artificial bee colony local optimization is used to tackle continuous attributes of the classification rule so as to determine the range of continuous attributes value.

In A_HACO algorithm, when a classification rule corresponding to the rule evaluation function is generated, artificial bee colony local optimization process will optimize the classification rule. In order to improve optimization efficiency, four classification rules corresponding to four candidate rule evaluation functions are optimized at the same time. The food source shown in Fig.1 is composed of continuous attributes from four classification rules.

Figure 1

Food source representation



Local optimization goal for classification problem is to make classification rule having higher rule evaluation function value, namely, maximizing the sum of four rule evaluation function value. The local optimization model is described by (10).

$$\text{Perf}=\max\sum_{i=1}^{4} Rf_i(r_i), \tag{10}$$

where $Rf_i()$ denotes the i-th rule evaluation function, r_i denotes the i-th classification rule.

In Ant colony optimization model for classification problems, continuous attributes are discretized. In artificial bee colony local optimization process, given the classification rule r_i (i=1, 2, 3, 4), the value constraint of continuous attribute a_{ij} in r_i needs to be converted into the form $[C_{ij}^{\min}, C_{ij}^{\max}]$, then initial population (namely *SN* food sources) is constructed, where *SN* denotes the number of employed bees or onlookers.

In artificial bee colony local optimization process, the food source X_i can be converted into four classification rules r_{ij} (j=1, 2, 3, 4). The probability with the food source X_i to be chosen by an onlooker is defined by (11).

$$P(X_i)=\frac{\sum_{j=1}^{4} Rf_j(r_{ij})}{\sum_{k=1}^{SN}\sum_{j=1}^{4} Rf_j(r_{kj})}. \tag{11}$$

Given the food source X_i, local search strategy to find a neighbor food source V_i for employed bees or onlookers is defined by (12).

$$\begin{aligned} &V_{ij}=[V_{ij}^{\min},V_{ij}^{\max}] \\ &\text{where,} \\ &V_{ij}^{\min}=X_{ij}^{\min}+\Phi_{ij}(X_{ij}^{\min}-X_{kj}^{\min}) \\ &V_{ij}^{\max}=X_{ij}^{\max}-\Phi_{ij}(X_{ij}^{\max}-X_{kj}^{\max}), \end{aligned} \tag{12}$$

where Φ_{ij} is a random number in the range of [-1,1], k is a random number different from i. The neighbor food source is abandoned if $V_{ij}^{\max}$ is lower than $V_{ij}^{\min}$. The algorithm compares V_i with X_i, if V_i has an equal or better nectar amount than X_i, then X_i is replaced by V_i in the memory. Otherwise, X_i is retained in the memory. In other words, a greedy selection mechanism is employed as the selection operation between the old and the neighbor food sources.

The food source X_i cannot be improved through the predetermined number of trials "*limit*", then X_i is abandoned by its employed bee and then the employed bee becomes a scout. The scout is responsible to search a new food source according to the formula (13).

$$\begin{aligned} &V_{kj}=[V_{kj}^{\min},V_{kj}^{\max}] \\ &\text{where,} \\ &V_{kj}^{\min}=X_{ij}^{\min}+rand(0,1)(X_{ij}^{\max}-X_{ij}^{\min}) \\ &V_{kj}^{\max}=X_{ij}^{\max}-rand(0,1)(X_{ij}^{\max}-X_{ij}^{\min}), \end{aligned} \tag{13}$$

where *rand*(0,1) is a random number between "0" and "1". New food source needs to satisfy $V_{kj}^{\min}$ lower than $V_{kj}^{\max}$.

2.2.3. Rule Set Comparing and Ants Redistributing

Rule set comparing module compares the quality of four rule sets corresponding to four candidate rule evaluation functions respectively, thus determines the quality of four candidate rule evaluation functions. More specifically, the precision of rule set is calculated firstly, rule evaluation function is ranked according to the precision of corresponding rule set. The precision is higher, rule evaluation function is ranked higher. The precision of rule set is defined by (14).

$$Cor(i) = \frac{p_i}{p_i + n_i}, \tag{14}$$

where, for the rule set corresponding to the rule evaluation function *i*, p_i and n_i refer to, respectively, the number of correctly and incorrectly covered examples.

Ants redistributing module dynamically adjusts the number of each group of ants according to the quality of rule evaluation functions. Rule evaluation functions with high quality are allocated more ants, and the number of ants corresponding to rule evaluation functions with low quality is decreased. In ants redistributing module, two variables are introduced, namely initial ant number *InitnbAnts* and ants adjusting factor *step*. In the beginning, each rule evaluation function has the same amount of ants, namely the number of ants $A(i)$ equal to *InitnbAnts*. Ants redistributing formula is defined as following.

$$A(i) = \begin{cases} A(i) + x \cdot step & A(i) + x \cdot step > 0 \\ 0 & A(i) + x \cdot step \le 0 \end{cases}, \tag{15}$$

where $i(1 \le i \le 4)$ is ID of rule evaluation function, x is a integer whose value is related to the rank of rule evaluation function, *step* is used to control the adjusting range of the number of ants. More specifically, x equals to 2 when the rule evaluation function is ranked first, x equals to 1 when the rule evaluation function is ranked second, x equals to -1 when the rule evaluation function is ranked third, x equals to -2 when the rule evaluation function is ranked fourth.

2.3. The A_HACO Algorithm Description

The A_HACO algorithm includes six key modules which are data preprocessing, rule generating, artificial bee colony optimization, rule set comparing, ants redistributing and convergence judging, respectively. Data preprocessing module charges to put training data into an array, and converts continuous attributes into categorical attributes. Rule generating module is used to construct a corresponding optimal classification rule for each candidate rule evaluation function. For the classification rule generated, artificial bee colony local optimization is used to tackle the continuous attributes so as to determine the range of continuous attributes value. Rule set comparing module compares the quality of four rule sets corresponding to four candidate rule evaluation functions respectively, thus determines the quality of four candidate rule evaluation functions. Ants redistributing module dynamically adjusts the number of each group of ants by formula (15) according to the quality of rule evaluation functions. Convergence judging module is used to judge whether the ending condition is satisfied or not, and return the optimal rule evaluation function and corresponding rule set.

The ending condition of A_HACO algorithm is that only one rule evaluation function is remained and the number of training samples uncovered is lower than the predefined threshold value.

3. Experimental Evaluations

In this part, the parameters tuning and the influence of artificial bee colony optimization process in the proposed algorithm are first discussed. All algorithms are implemented in C language and executed on a Core(i7), 2.93GHZ, 3GB RAM computer.

To show the performance, the proposed A_HACO algorithm is then compared with the related algorithms on six public test instances [7]. Test dataset is described as in Table 1. In test instances “SONAR” and “Balance”, all attributes are continuous. In test instances “TTT” and “CKR”, all attributes are categorical. In test instances “CMC” and “Flags”, there are both continuous attributes and categorical attributes.

The dataset is divided into two parts, 2/3 dataset is used as training set and the remained 1/3 dataset is used as testing set.

Table 1
Test dataset introduction

test instance	attribute type	samples number	attribute number
SONAR	Continuous	208	60
Tic-Tac-Toe TTT	Categorical	958	9
Balance	Continuous	625	4
Contraceptive Method Choice CMC	Hybrid	1473	9
Chess-King-Rook CKR	categorical	3196	36
Flags	Hybrid	194	30

3.1. Parameter Tuning

In A_HACO algorithm, there are some parameters needing to be discussed which are the parameters (τ_{max}, τ_{min}, *a*, *b* and *r*) related to ACO, the parameters (ω, γ and *m*) related to rule evaluation function, the parameters (*limit* and *MCN*) related to ABC and the parameters *step* related to ants redistributing.

Initial pheromone value is used to make the algorithm to find solutions without the preference information in the beginning. With the running of the algorithm, the quality of solutions are different, and the pheromone increment of corresponding edges are different. The ants always search towards the direction with high pheromone value. Therefore, the performance of the algorithm is related to the pheromone increment value not initial pheromone value (or the upper and lower bound of pheromone value). Thus, the parameters τ_{max} and τ_{min} are not discussed in this paper. The configuration of parameters(ω, γ and *m*) related to rule evaluation function are discussed in [11], and the better settings for these parameters are ω equal to 0.40, γ equals to 0.25 and *m* equal to 4. The parameters *limit* and *MCN* related to *ABC are set as* 20 and 2000 respectively according to the experience. The ants colony size *InitnbAnts* is set as 50, and the artificial bee colony size *SN* is set as 20. Except for the above parameters, other parameters are more complex and sensitive in this algorithm. Their ranges are shown in Table 2.

To set appropriate values for these parameters, we tune them in the sequential orders. For parameter *a*, we vary its value one at a time, while setting the values of the other parameters to their default values. For the next untuned parameter *b*, vary its value one at a time while setting the values of tuned parameters to the obtained most appropriate ones and the values of the other untuned parameters to their default values. Then the other two parameters are tuned in the same way as *b*. The termination condition of this algorithm is that only one rule evaluation function is remained (others are abandoned) and the number of samples uncovered is lower than 5%. The A_HACO algorithm is run ten times on the test selected instances, and the average of running results is final results shown in Figure 2.

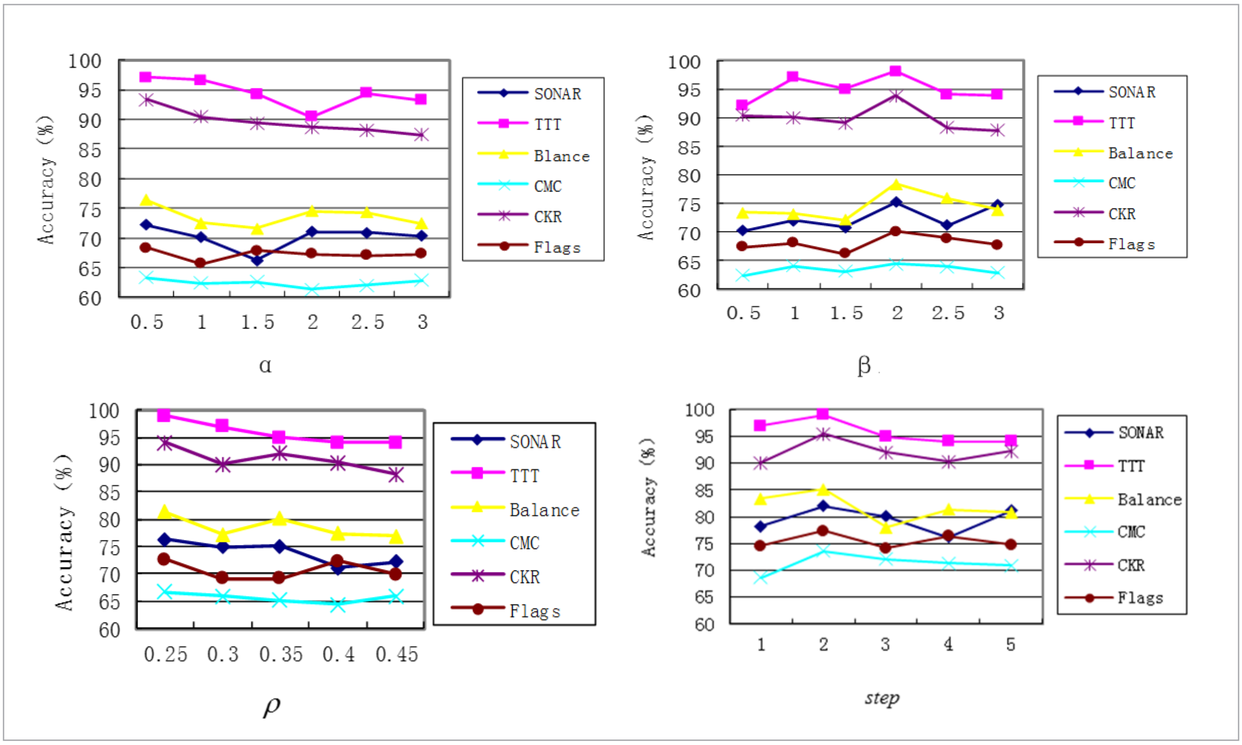
Table 2
The tuned parameters

Parameter	Default	Range
a	1.75	From 0.50 to 3.00 with increment 0.50
b	1.75	From 0.50 to 3.00 with increment 0.50
r	0.35	From 0.25 to 0.45 with increment 0.05
step	3	From 1 to 5 with increment 1

We can see that the performance of algorithm proposed varies with different parameter configuration. The comparatively better settings for these parameters are *a*=0.5, *b*=2.0, *r*=0.25 and *step*=2.0 for the algorithm proposed.

Figure 2

The result of parameter Setting for *a, b, r* and *step*



3.2. The Influence Analysis of Artificial Bee Colony Search on Algorithm Performance

To test the influence of artificial bee colony local search on algorithm performance, we compare the running results of the adaptive ant colony optimization algorithm without artificial bee colony local search called A_ACO and A_HACO algorithm. The parameters and the termination condition of the two algorithms are set as in Section 3.1. The accuracy (Acc) and running time (RT) of algorithms are shown in Table 3, "REF" denotes the rule evaluation function selected by A_HACO.

The ACO algorithm can only tackle categorical attributes. Continuous attributes need to be discretized in preprocessing. However, the discretization operation of continuous attributes will influence the performance of algorithms. In A_HACO algorithm, after a classification rule is generated, artificial bee colony

Table 3

Accuracy comparison between A_ACO and A_HACO

dataset / algorithm	SONAR		Tic-Tac-Toe		Balance		Contraceptive Method Choice		Chess-King-Rook		Flags	
	Acc (%)	RT (s)	Acc (%)	RT (s)	Acc (%)	RT (s)	Acc (%)	RT (s)	Acc (%)	RT (s)	Acc (%)	RT (s)
A_ACO	75	105.034	99.27	75.154	79.68	74.213	69.69	83.216	94.17	96.620	70.62	99.238
A_HACO	82.56	172.352	99.28	103.915	85.74	108.308	69.68	127.425	95.63	139.413	78.26	150.723
REF	Klosgen		Q^+		M-estimate		Klosgen		F-measure		Klosgen	

local optimization is used to tackle continuous attributes of the classification rule so as to determine the range of continuous attributes value. From Table 3, we can see that the accuracy of A_HACO algorithm is better than the A_ACO algorithm. However, the running time of A_HACO algorithm is longer than A_ACO algorithm because of the artificial bee colony local search. Classification algorithm is off-line learning process, therefore, it is acceptable that running time of the algorithm is increased to some extent. At the same time, the experiment results of A_HACO algorithm reveal that the rule evaluation function selected varies with test instance.

3.3. The Performance Comparison of Classification Algorithms

In this part, we compare the A_HACO algorithm with the classification algorithm with incomprehensible classifier Kalman Filter+ SVM [4], and the classification algorithms with comprehensible classifier WrapTree [20], Antminer+ and PSO/ACO2 on the test instances. In order to illustrate algorithm performance varying with rule evaluation function, this paper tests the performance of Antminer+ algorithm with Klosgen, F-measure, M-estimate and Q^{+}, respectively. The parameters and the termination condition of the A_HACO algorithm are set as in Section 3.1. The results of the experiment are shown in Table 4.

From Table 4, we can see that the performance of Antminer+ algorithm varies with rule evaluation on the given test instance, and there is no rule evaluation function which can perform well on all test instances. From the overall running results, the performance of Kalman Filter+ SVM algorithm is better than other algorithms on the test instance Balance, and the accuracy of A_HACO is better than other algorithms on the test instances SONAR, Contraceptive Method Choice and Flags, A_HACO and Antminer+ algorithms have nearly the same performance on the test instance Tic-Tac-Toe, and A_HACO, PSO/ACO2 and Antminer+ (M-estimate) algorithms have nearly the same performance on the test instance Chess-King-Rook. Hence, among the classification algorithms with comprehensible classifier, the performance of A_HACO is better than others, which is owed to the reason that the A_HACO algorithm can adaptively select the appropriate rule evaluation function according to the data set given and tackle both categorical and continuous attributes.

Table 4

Classification algorithm accuracy Comparison(%)

dataset / algorithm	SONAR	Tic-Tac-Toe	Balance	Contraceptive Method Choice	Chess-King-Rook	Flags
A_HACO	82.56	99.28	85.74	74.68	95.63	78.26
Antminer+ (Klosgen)	68.27	99.27	69.28	58.45	93.71	65.98
Antminer+ (F-measure)	72.12	99.16	70.08	56.82	91.71	61.34
Antminer+ (M-estimate)	70.67	98.54	75.20	61.64	94.43	60.82
Antminer+ (Q^{+})	71.63	99.27	51.20	63.61	89.27	62.89
PSO/ACO2	77.08	99.35	82.71	74.58	93.46	64.25
WrapTree	70.22	83.68	76.34	59.62	90.24	61.36
Kalman Filter+ SVM	73.68	91.02	89.72	62.23	91.67	64.56

4. Conclusions

To address the classification problems with hybrid attributes and get the comprehensible classification rules, an adaptive hybrid ant colony optimization algorithm called A_HACO is proposed. Given the data set, the algorithm can adaptively select the appropriate rule evaluation function, which can effectively improve the classification accuracy. The artificial bee colony local search strategy makes the proposed algorithm can tackle not only the categorical attributes but also continuous attributes. The experimental results reveal that the algorithm proposed has better accuracy and applicability. In the future, we will focus on improving the ant redistribution mechanism to further improve the efficiency of A_HACO.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities (N171604009). This work was funded by the Chinese State Foundation for Overseas Study.

References

1. Abaszade, M., Effati, S. Stochastic Support Vector Machine for Classifying and Regression of Random Variables. Neural Processing Letters, 2017, 1-29. https://doi.org/10.1007/s11063-017-9697-0
2. Abpeykar, S., Ghatee, M. An Ensemble of RBF Neural Networks in Decision Tree Structure with Knowledge Transferring to Accelerate Multi-classification. Neural Computing & Applications. 2018. https://doi.org/10.1007/s00521-018-3543-9
3. Carvalho, A. B. D., Savegnago, T., Pozo, A. Swarm Intelligence for Rule Discovery in Data Mining. Proceedings of the International Conference on Enterprise Information Systems, Madeira, Portugal, 2010, June 8-12, 314-319.
4. Hassan, M., Isa, D., Rajkumar, R. Reducing Support Vector Machine Classification Error by Implementing Kalman Filter. International Journal of Intelligent Systems & Applications, 2013, 5(9), 10-18. https://doi.org/10.5815/ijisa.2013.09.02
5. Holden, N., Freitas, A. A Hybrid Particle Swarm/Ant Colony Algorithm for the Classification of Hierarchical Biological Data. IEEE Swarm Intelligence Symposium, Pasadena, USA, 2005, June 8-10, 100-107.
6. Holden, N., Freitas, A. A. A Hybrid PSO/ACO Algorithm for Discovering Classification Rules in Data Mining. Journal of Artificial Evolution and Applications, 2008, 1-12. https://doi.org/10.1155/2008/316145
7. Hettich, S., Bay, S. D. The UCI KDD Archive. Irvine, CA: University of California, Department of Information and Computer Science, 1999. http://kdd.ics.uci.edu.
8. Lissovoi, A., Witt, C. Runtime Analysis of Ant Colony Optimization on Dynamic Shortest Path Problems. Theoretical Computer Science, 2015, 561, 73-85.https://doi.org/10.1016/j.tcs.2014.06.035
9. Liu, J., Zhu, H., Ma, Q. An Artificial Bee Colony Algorithm with Guide of Global & Local Optima and Asynchronous Scaling Factors for Numerical Optimization. Applied Soft Computing, 2015, 37(C), 608-618. https://doi.org/10.1016/j.asoc.2015.08.021
10. Martens, D., Backer, M. D. Classification with Ant Colony Optimization. IEEE Transactions on Evolutionary Computation, 2007, 11(5), 651-665. https://doi.org/10.1109/TEVC.2006.890229
11. Minnaert, B., Martens, D., Backer, M. D. To Tune or Not to Tune: Rule Evaluation for Metaheuristic-based Sequential Covering Algorithms. Data Mining & Knowledge Discovery, 2015, 29(1), 237-272. https://doi.org/10.1007/s10618-013-0339-5
12. Michalak, M., Sikora, M., Wrobel, L. Rule Quality Measures Settings in a Sequential Covering Rule Induction Algorithm-an Empirical Approach. Computer Science and Information Systems, Lodz, Poland, 2015, September 13-16, 109-118. https://doi.org/10.15439/2015F388
13. Mencía, E. L., Janssen, F. Learning Rules for Multi-Label Classification: A Stacking and a Separate-and-Conquer Approach. Machine Learning, 2016, 105(1), 1-50. https://doi.org/10.1007/s10994-016-5552-1
14. Otero, F. E. B., Freitas, A. A., Johnson, C. G. A New Sequential Covering Strategy for Inducing Classification Rules with Ant Colony Algorithms. IEEE Transactions on Evolutionary Computation, 2013, 17(1), 64-76. https://doi.org/10.1109/TEVC.2012.2185846
15. Oyeleke, R. O., Yu, C., Chang, C. K. Situ-Centric Reinforcement Learning for Recommendation of Tasks in

Activities of Daily Living In Smart Homes. IEEE 42nd Annual Computer Software and Applications Conference, Tokyo, Japan, 2018, July 23-27, 317-322. https://doi.org/10.1109/COMPSAC.2018.10250

16. Pourpanah, F., Lim, C. P., Saleh, J. M. A Hybrid Model of Fuzzy ARTMAP and Genetic Algorithm for Data Classification and Rule Extraction. Expert Systems with Applications, 2016, 49(C), 74-85. https://doi.org/10.1016/j.eswa.2015.11.009
17. Parpinelli, R. S., Lopes, H. S., Freitas, A. A. An Ant Colony based System for Data Mining: Applications to Medical Data. International Conference on Genetic and Evolutionary Computing, San Francisco, USA, 2001, July 7-11, 791-797.
18. Połap, D., Woźniak, M., Napoli, C., Tramontana, E., Damaševičius, R. Is the Colony of Ants Able to Recognize Graphic Objects? The Fifth International Conference on Information Science and Technology, Druskininkai, Lithuania, 2015, October 15-16, 376-387. https://doi.org/10.1007/978-3-319-24770-0_33
19. Salama, K. M., Abdelbar, A. M. Learning Cluster-based Classification Systems with Ant Colony Optimization Algorithms. Swarm Intelligence, 2017, 11(3-4), 1-32. https://doi.org/10.1007/s11721-017-0138-5
20. Somvanshi, M., Chavan, P. A Review of Machine Learning Techniques Using Decision Tree and Support Vector Machine. International Conference on Computing Communication Control and Automation, Pune, India, 2017, August 12-13. https://doi.org/10.1109/ICCUBEA.2016.7860040
21. Shah, H., Ghazali, R., Nawi, N. M. Hybrid Global Artificial Bee Colony Algorithm for Classification and Prediction Tasks. Journal of Applied Sciences Research, 2013, 9(5), 3328-3337.
22. Salhi, S. Handbook of Metaheuristics (2nd edition). Journal of the Operational Research Society, 2014, 65(2), 320-320. https://doi.org/10.1057/jors.2013.151
23. Tsuboi, K., Suga, S., Kurihara, S. Hierarchical Pattern Mining Based on Swarm Intelligence. Genetic and Evolutionary Computation Conference Companion, Berlin, Germany, 2017, July 15-19, 47-48. https://doi.org/10.1145/3067695.3082038
24. Woźniak, M., Połap, D. Bio-Inspired Methods Modeled for Respiratory Disease Detection from Medical Images. Swarm and Evolutionary Computation, 2018, 41, 69-96. https://doi.org/10.1016/j.swevo.2018.01.008