Journal of Computer Science

Original Research Paper

Hybrid Ant Colony Optimization and Genetic Algorithm for Rule Induction

[1]Hayder Naser Khraibet AL-Behadili, [2]Ku Ruhana Ku-Mahamud and [3]Rafid Sagban

[1]Department of Computer Science, Shatt Alarab University College, Basra, Iraq
[2]School of Computing, Universiti Utara Malaysia, Kedah, Malaysia
[3]Department of Software, University of Babylon, Babylon, Iraq

Article history
Received: 12-04-2020
Revised: 06-07-2020
Accepted: 23-07-2020

Corresponding Authors:
Hayder Naser Khraibet Al-Behadili
Department of Computer Science, Shatt Alarab University College, Basra, Iraq
Email: haider872004@gmail.com

Abstract: In this study, a hybrid rule-based classifier namely, ant colony optimization/genetic algorithm ACO/GA is introduced to improve the classification accuracy of Ant-Miner classifier by using GA. The Ant-Miner classifier is efficient, useful and commonly used for solving rule-based classification problems in data mining. Ant-Miner, which is an ACO variant, suffers from local optimization problem which affects its performance. In our proposed hybrid ACO/GA algorithm, the ACO is responsible for generating classification rules and the GA improves the classification rules iteratively using the principles of multi-neighborhood structure (i.e., mutation and crossover) procedures to overcome the local optima problem. The performance of the proposed classifier was tested against other existing hybrid ant-mining classification algorithms namely, ACO/SA and ACO/PSO2 using classification accuracy, the number of discovered rules and model complexity. For the experiment, the 10-fold cross-validation procedure was used on 12 benchmark datasets from the University California Irwine machine learning repository. Experimental results show that the proposed hybridization was able to produce impressive results in all evaluation criteria.

Keywords: Rules-based Classification, Swarm Intelligence, Machine Learning, Data mining, Ant-Miner

Introduction

Ant Colony Optimization (ACO) is a metaheuristic framework that handles combinatorial optimization and other problems (Jabbar *et al*., 2020; López-Ibáñez *et al*., 2016; Sagban *et al*., 2016). The inspiring source of ACO is derived from actual behavior of ants and artificial pheromone trails used as indirect communication medium. Heuristic information and pheromone trails are two types of numerical information employed by ACO. The heuristic information is taken from the problem being solved. Pheromone trails represent a distributed, numerical information that the ACO adapt during its execution to reflect the search experience (Dorigo and Stützle, 2004). Many applications involve the usage of ACO metaheuristic framework such as scheduling (Blum, 2005), travel salesman problem (Sagban *et al*., 2017), assembly line balancing (Blum, 2008), sequential ordering (Dorigo and Stützle, 2010), DNA sequencing (Blum *et al*., 2008), packet-switched routing (Di Caro and Dorigo, 1998), feature selection (Kanan *et al*., 2007), data clustering (Jabbar and Ku-Mahamud, 2018; Jabbar *et al*., 2019a; 2019b) and data classification (Al-Behadili *et al*., 2019; 2018b; 2018a).

Ant-Miner classifier is a prominent ACO variant for data mining classification task. It generates a set of rules from the data to classify them into predetermined classes (Al-Behadili *et al*., 2020a). These rules are used to predict the unknown classes for unseen data. The Ant-Miner generates rules in the form of if–then rules that are considered simple and comprehensible knowledge representor. Examples of Ant-Miner applications include medical diagnosis, bankruptcy determination and stock price prediction (Al-Behadili, 2018; Ripon, 2019).

The hybrid classifier in the Ant-mining literature introduced by (Saian and Ku-Mahamud, 2012) proposed Ant-Miner coupled with Simulated Annealing (SA) to generate a list of classification rules. In Ant-Miner, each ant discovers a rule. The study proposed SA as a local search procedure to improve this rule iteratively. The SA works for each rule on the basis of the temperature variable, which starts at a high value and then decreases on the basis of predefined factors. The search runs a certain number of iterations and selects the best rule from the available neighbourhood depending on its quality. Using the SA mechanism, which starts with high temperatures, allows the rule with low quality to be selected. Then, temperature will be decreased and the difference between the current and previous qualities will be crucial for selecting the rule with high quality. The performance matrix is indicated on the basis of the rule quality, the number of discovered rules and the terms per rule. The performance of this approach was tested using 13 datasets from a UCI repository, showing that it is comparable with the original Ant-Miner in the predicative accuracy.

ACO/PSO2 is a hybrid swarm intelligence metaheuristic algorithm for rule-based classification that combined ACO with Particle Swarm Optimization (PSO) (Holden and Freitas, 2008). The pruning procedures of ACO/PSO2 are applied to discover the best rule from each iteration. ACO/PSO2 uses two pruning procedures. The first procedure is the original Ant-Miner pruning procedure and applied to the best rule discovered whose number of terms is less than a fixed value (i.e., 20). If the constructed rule entails more than 20 terms for each rule, then the pruning iterates to remove the unimportant or detrimental terms from the classification rule until the number is decreased to 20 terms. Subsequently, the Ant-Miner pruning procedure is implemented.

However, the Ant-Miner classifier suffers from premature exploitation because of the absence of any local search in its structure. Ant-Miner is not designed to explore the neighborhoods of the current rule and does not consume more time in improving it iteratively. The neighborhood structures are not fully covered. In this manner, this type of search is over-explorative because it is either a single neighborhood structure movement, as exemplified in (Saian and Ku-Mahamud, 2012), or it does not profit from local search (Holden and Freitas, 2008; Martens *et al*., 2011). Therefore, various neighborhood structures can be developed to catapult the search to another point, which makes it possible to completely utilize the neighborhood. The present study proposes the hybridization of Genetic Algorithm (GA) with Ant-Miner classifier algorithm for a more mature exploitation.

This paper is constructed as follows. The next section illustrates the proposed hybrid ACO/GA classifier. The materials and methods section present experiments method, performance evaluation metrics, databases, classifiers and the parameter setting used in our experiments. Then, discusses the experimental results of the proposed classifier. Finally, provides the conclusions and suggestions for possible future research.

Hybrid ACO/GA algorithm

The overall goal of ACO/GA is to benefit from the characteristics to form the neighborhood structures. In GA, a strong inter-correlation exists between exploration components (e.g., mutation) and exploitation components (e.g., crossover). Furthermore, GA has succeeded in using the crossover and mutation operators in utilizing multi-neighborhood structures. Figure 1 shows how GA utilizes crossover and mutation to improve the search in two of its neighborhood structures in the proposed hybridization.

The ACO/GA classifier begins to construct one classification rule from the training dataset. This discovered rule is then inserted in the rule list, in which every case in the data that satisfies this rule is removed from the training dataset. These operations terminate when all the cases in the training database are lower than the pre-specified constant values known as UncoveredInstancesNO.

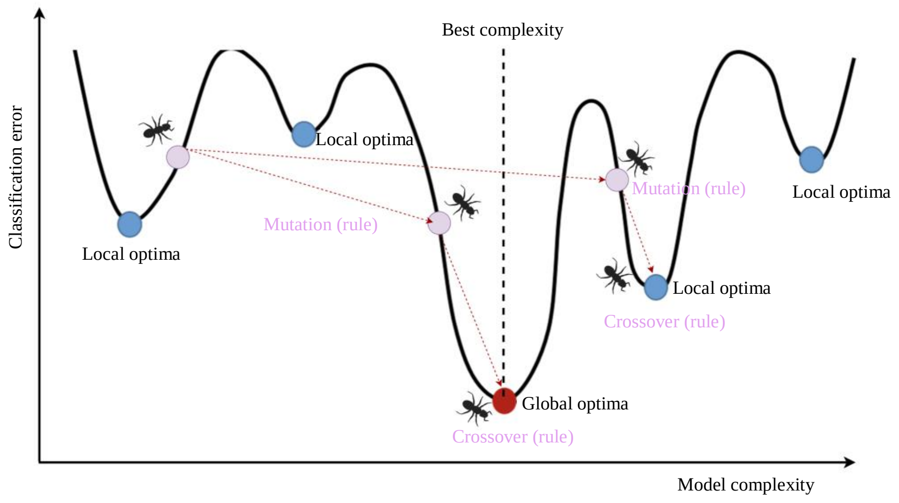

Fig. 1: Proposed hybrid ACO/GA search strategy

This approach has three major stages namely, rule building *(RuleConstructs)*, pruning rule *(RulePrune)* and updating pheromone. The initial procedure is the *RuleConstructs*, where every ant begins to allow terms to inserted in the rule. The ant inserts a term at a time while raise the classification accuracy according to its probability value. The probability function that is use to add term to the current rule is expressed in Equation 1 as follows (Parpinelli *et al*., 2002):

$$Probability = \frac{\left[\tau_{ij(t)}\right]\left[\eta_{ij}\right]}{\sum_{i=1}^{a} xi \cdot \sum_{j=1}^{bi}\left[\tau_{ij(t)}\right]\left[\eta_{ij}\right]} \quad (1)$$

where, [$\tau_{ij(t)}$] is the amount of pheromone concentration for each term at iteration (t); [η_{ij}] is the problem depending upon heuristic function; a is the attribute number in the dataset; bi is the term number for each attribute; and xi is set to 1 (if the attribute is not yet visited by the ant) and 0 (otherwise). In addition, the heuristic function value is used together with the pheromone value to decide on the term selection. In ACO/GA, the heuristic function is inspired by information theory. The ACO/GA computes the amount of information contained in each term (entropy). The heuristic function can be expressed as follows (Parpinelli *et al*., 2002):

$$\eta_{ij} = \frac{\log_2 k - H\left(W \mid A_i = V_{ij}\right)}{\sum_{i=1}^{a} xi \cdot \sum_{j=1}^{bi}\left(\log_2 k - H\left(W \mid A_i = V_{ij}\right)\right)} \quad (2)$$

$$H\left(W \mid A_i = V_{ij}\right) = -\sum_{w=1}^{k}\left[\frac{P\left(W \mid A_i = V_{ij}\right)}{T_{ij}}\right] * \log_2\left[\frac{P\left(W \mid A_i = V_{ij}\right)}{T_{ij}}\right] \quad (3)$$

Where:

w	=	The class attribute
k	=	Represent the classes number
$P(W\|A_i = V_{ij})$	=	The partition containing the instances, where the feature A_i has a value of V_{ij} with class w
$\|T_{ij}\|$	=	Represents the total of instances number in partition T_{ij} (instances where attribute A_i has a value of Vij)
a	=	Represents the total number of attributes
b_i	=	The number of values in the particular attribute i

This process is repeated to complete the construction rule based on two conditions. The first one, the current rule is not covered the prespecified the minimum instances by rule; while the second condition; certain attributes from the data are not yet used by the current constructed rule. Once the rule is finished, the classifier assigns the Then part of the rule by selecting the majority class vale among the all instances covered by the discovered rule. The discovered rule will then undergo pruning procedure *RulePrune*, which aims to avoid the overfitting problem by reducing the length of the constructed rules to increase simplicity. The procedure deletes one term from the rule at a time as the classification quality of the constructed rule improves. This method loop until only one term is left in the pruned rule or just no more improvement happens. The class value of the rule will have a high chance to be changed through this stage due to the fact that the pruned rule may cover other instances from dataset compared with the cases covered by the original one. The pheromone is updated after rule construction and prune procedures. Updating the pheromone has two basic stages. Firstly, the amount of pheromone amount is increased in all terms, including the discovered rule, according to rule quality function in Equations 4 and 5 as follows (Parpinelli *et al*., 2002):

$$\tau_{ij(t+1)} = \tau_{ij(t)} + \tau_{ij(t)} \cdot Q \quad (4)$$

$$Q = +\frac{TP}{TP + FN} * \frac{TN}{FP + TN} \quad (5)$$

where, *TP* represents the cases number covered and classified by the constructed rule, *FN* represents the number of cases covered by constructed rule but it has a different class, *TN* represents the cases number that not covered by the constructed rule and has a class different from that classified by the constructed rule and *FP* represents the cases not covered by the constructed rule but has a class classified by the rule.

Secondly, evaporating each term does not appear in the rule to avoid cumulative pheromone in un important terms. Then, another ant constructs its rule derived from pheromone history. The process is finished based on the following stopping conditions. The first condition is that the number of discovered rules must be equal to the number of ants. The second condition is according to the number of rule convergence that is statically determined, in which the ant starts to converge by building a rule similar to that previously constructed. The best quality rule from all discovered rules will be added to the discovered rule list.

To speed up the search in the proposed ACO/GA, GA is only applied for the iteration-best rule instead of applying it for all constructed rules to ensure that the algorithm stays lightweight. The change that may be applied to the classification rule is defined by a neighborhood structure. A neighborhood structure is a function N: $S \rightarrow 2S$ that assigns a set of neighborhood

$N(s) \subseteq S$ to every $s \in S$. $N(s)$ is also called the neighborhood of s. The best rule in discovered-rule-list will be treated by crossover and mutation procedures. Therefore, the best discovered rule will be kernelled to form the neighborhood structure.

ACO/GA is a multi-neighborhood structure algorithm. In each iteration, the mutation generates a new starting rule where the crossover can be applied. ACO/GA selects two parent chromosomes (rules). The iteration-best rule represents the FirstParent. The SecondParent is a random classification rule selected from the classification list of discovered rules. Then, the mutation operator is used to maintain genetic diversity from one generation of a rule to the next. The mutation rate parameter is performed to decide if the rules should have a mutation. The parameter of the mutation rate is compared with a random number to perform the mutation operation. The mutation operator selects the random bits in the parent chromosomes and flips the value of this bit. Thus, the size of mutation is 2-terms exchange. Figure 2 shows the pseudocode of the mutation operator.

The crossover operator is the process in which parent chromosomes (rules) exchange genetic information to create the best rule. The crossover rate parameter is used to perform crossover in a similar approach to the mutation operator. If the crossover rate is greater than the random number, then the crossover will be implemented. Different methods are used to trade genetic information between two individuals. The crossover operation used in this study guarantees a fully matured exploitation for such promising search regions. The length of crossover movement CrossOverLength in ACO/GA is set to half of the classification rule. Thus, the crossover terms are replaced up to two CrossOverLength from the FirstParent and SecondParent. Figure 3 shows the pseudocode of the crossover method.

The procedure then repeatedly performs operations from the given rule until no further improvement can be achieved. The acceptance criterion used to accept the better-quality classification rule during the crossover and mutation stages can be expressed in Equations 6 and 7:

$$Acceptance\ Criterion = \begin{cases} BestRule^{*}, if\ Quality\left(BestRule^{*}\right) > Quality\left(BestRule\right) \\ BestRule, Otherwise \end{cases} \quad (6)$$

$$Quality\left(BestRule^{*}\right) - \frac{TP}{TP+FN+FP} + \frac{TN}{FP+TN} \quad (7)$$

where, *TP* represents the cases number covered and classified by the constructed rule, *FN* represents the number of cases covered by constructed rule but it has a different class, *TN* represents the cases number that not covered by the constructed rule and has a class different from that classified by the constructed rule and *FP* represents the cases not covered by the constructed rule but has a class classified by the rule. The acceptance criteria in GA will be the rudder that guides the search towards the bottom of the shape of the neighbourhood structure. Furthermore, it determines the size of movement in the current search region.

```
Mutation pseudocode

IF MutationRate > Random();
    MutatedFirstTerm = SelectMutatedTerm();
    MutatedSecondTerm = SelectMutatedTerm();
    Offspring = Mutation (MutatedFirstTerm, MutatedSecondTerm, FirstParent, SecondParent);
ELSE: Offspring= (FirstParent, SecondParent);
END IF
```

Fig. 2: Mutation operator pseudocode

```
Crossover psaeudocode

    IF CrossoverRate > Random();
  Offspring = Crossover (FirstParent, SecondParent, CrossOverLength, Rule.length);
         For (int i = CrossOverLength; i<= Rule.length; i++)
              {
              int  temp = FirstParent[i];
                   FirstParent[i] = SecondParent [i];
                   SecondParent [i] = temp;
              }
         END Loop
  ELSE: Offspring= (FirstParent, SecondParent);
  END IF
```

Fig. 3: Crossover operator pseudocode

```
ACO/GA algorithm

Input: arff data set
Output: classification rule-list
TrainingDataSet= {all TrainingData instances};
Initialization of ConstructionRuleList =[];
WHILE (TrainingDatset>Maximum UncoveredInstancesNO)
AntIndexNO =1; ConvergenceTestNO=1;
PheromoneInitialization;
REPEAT
RuleConstructs;
RulePrune;
UpdatePheromone;
IF(Current ConstructedRule = Previous ConstructedRule)
THEN ConvergenceTestNO = ConvergenceTestNO + 1;
ELSE ConvergenceTestNO = 1;
END IF AntIndexNO = AntIndexNO + 1;
UNTIL(AntIndexNO >= AntNumber) OR (ConvergenceTestNO >=RuleConvergenceNo)
SelectTheBestRule;
        REPEAT
           BestRule′ = Mutation (BestRule);
           BestRule* =Crossover(BestRule');
           BestRule = AcceptanceCriterion (BestRule, BestRule*);
         UNTIL termination condition met
Insert BestRule to ConstructionRuleList;
TrainingData instances = TrainingData instances -{InstancesSetCoveredByRule};
END WHILE
```

Fig. 4: ACO/GA algorithm pseudocode

The ACO/GA pseudocode shows the adaptation of the aforementioned components of GA-based algorithm (i.e., crossover, mutation and acceptance criteria) in the Ant-Miner framework (Fig. 4). The combined feature makes the proposed classifier substantially different from the previous Ant-Mining classification algorithm.

Materials and Methods

Experiments

A 10-fold cross-validation method is used in our experiments. In this method, the dataset is split into 10 subsets. Each subset is equally sized, where nine are used for the training process. The remaining subset is used in the testing stage. This process is repeated 10 times with a different subset for training and testing to ensure that all subsets are used for training and testing. Subsequently, the performance of all folds is averaged and the standard deviations are calculated. The 10-fold cross-validation method has been also adopted in other ant-mining classifier studies (Al-Behadili *et al*., 2020b; Parpinelli *et al*., 2002; Saian and Ku-Mahamud, 2012).

Performance Evaluation

The evaluation in this study is performed on the basis of three criteria. The first criterion is the classification accuracy in discovering the rule list, which is called the correct classification rate. This criterion is based on the correctly classified instances in the test data. Each time, the training subsets consist of n number of instances. The classifier constructs the training and test subsets that will be used to test the performance. The correct classification instances will determine the performance of the proposed classifier. The second criterion is the size in discovering the rule list, which is measured by the total number of discovered rules. The third criterion is the model size, which is measured by the amount of terms per rule. The number of terms (conditions) refers to the number of antecedents carried by each rule.

This study also computes for the classification accuracy rank and the model size rank of the statistical results, in which the nonparametric Friedman test is conducted with the Holm post-hoc test. In this manner, the performance of all classifiers in accordance with classification accuracy and simplicity can be observed. Then, the result of the nonparametric Friedman test with Holm’s post hoc test is used to determine the average classification accuracy rank versus the average number of discovered rule rank and the average model size rank for all classifiers. This test aims to find the optimal classifier that balances different objectives. The test is conducted to rank the algorithms’ performance for each dataset in descending order. A low rank implies good algorithm performance. The test is used to rank the best classifier that balances between classification accuracy and model size. The A classifier dominates B classifier if and only if the following two (2) conditions are true: The first condition, A is not worse than B with respect to both objectives, i.e., classification accuracy and the model size. The second condition, A is strictly better than B at least in one (1) objective. Thus, A classifier is becoming optimal only if and only if it is not dominated by any other classifiers (Dua and Karra, 2017).

Databases

Several experiments are performed using 12 UCI benchmark datasets in Weka's ARFF format to test the performance of the adaptive algorithm (Dua and Karra, 2017). These datasets are famous in the ant-mining algorithms literature and demonstrate different attribute numbers, which lie between 4 and 22. The attributes exhibit categorical and continuous styles. The datasets also differ in instance size number within the range of between 148-8124. The main descriptions of the experimental datasets are listed in Table 1.

Classifiers

The implementation of hybridizing Ant-Miner with GA or ACO/GA is evaluated with two other hybrid classifiers, namely, ACO/SA and ACO/PSO 2. These classifiers are considered the most related classifiers in ant-mining literature. The ACO/PSO 2 classification rule discovery software packages is publicly available on: https://sourceforge.net/projects/psoaco2/.

In other hand, the ACO/SA and ACO/GA are implemented based on the available open-source software packages of Ant-Miner classifier: https://sourceforge.net/projects/guiantminer.

Saian and Ku-Mahamud (2012) combined ACO with SA, each ant discovers one best rule according to the hybrid ACO with SA algorithm. The SA algorithm can find an optimal solution for local optimisation problems. It is dependent on a variable named temperature (Aarts *et al.*, 1997). The proposed hybrid algorithm uses the iteration on the temperature variable by starting with a high value. Therefore, in the beginning, all rules have the same probability to be selected. Then, temperature reduces and all rules will have a chance to be selected as the best rule. In this manner, the algorithm can avoid the local optimisation problem of the solution. Thereafter, the best rule in the iteration will be selected and the best among all iterations will then be added into the rule set. A successful hybridisation between ACO and PSO algorithm has been achieved by (Holden and Freitas, 2008) that introduced a new hybrid ACO and PSO2 algorithm for the discovery of classification rules. This classification algorithm can directly cope with discrete and continuous attributes.

Parameter Setting

The values of Ant-Miner classifiers parameter are adopted from (Robu *et al.*, 2015; López-Ibáñez *et al.*, 2016; Raymer *et al.*, 2000) to ensure a fair evaluation of the results, all classifiers use the same values. Table 2 shows the parameters values used in the experiment.

Table 1: The datasets characteristics

	Description		
The name of datasets	Number of attributes	Number of instances	Number of classes
Scale Balance	4	625	3
Ljubljana Breast Cancer	9	286	2
Wisconsin Breast Cancer	9	699	2
CreditA	15	690	2
Diabetes Data	8	768	2
Cleveland Heart disease	13	303	5
Statlog Heart disease	13	270	2
Iris dataset	4	150	3
Medical Lymphography	18	148	4
Mushroom Data	22	8124	2
Segment Data	19	2310	7
Vehicle Data	18	846	4

Table 2: Experimental parameters

Parameter	Description	Value
AntNumber	The number of ants used to discover rule	10.0
MCR	The mini number of instances covered by rule	5.0
UncoveredInstancesNO	The max number of uncovered instances by the rule	10.0
RuleConvergenceNo	Rules Convergence Number	10.0
NI	iterations Number	10.0
CR	Crossover Rate	0.8
MR	Mutation Rate	0.1

Experimental Results

This section compares the results of the ACO/GA classifier with those of related classifiers with different rule pruning procedures. These classifiers include the ACO/PSO2 and ACO/SA. Experiments on 12 datasets from the UCI repository are conducted for all classification algorithms. In the first evaluation method, Table 3 shows the experimental results of the average classification accuracy. The result presents the average classification accuracy and the numbers after the symbol '+/−' are standard deviations. For each dataset, the best result is written in bold. Table 3 shows that the ACO/GA is better than ACO/PSO2 in 11 datasets. The ACO/GA is better than ACO/SA in eight datasets. Among all classifiers, the ACO/GA achieves the highest results in eight datasets. The ACO/SA obtains the second-best performance in three datasets [i.e. Wisconsin Breast Cancer, Cleveland Heart disease and Vehicle Data], whereas the ACO/PSO2 obtains the best results in only one dataset. Table 4 shows that the ACO/GA has the lowest number of discovered rules in all datasets for 10-fold cross-validation compared with ACO/PSO 2. The ACO/GA obtains the lowest results in 11 datasets compared with the ACO/SA. Table 5 shows that the ACO/GA achieves the best results for model size in 11 datasets compared with the ACO/PSO 2. By using the same token, the ACO/GA achieves the best results in 11 datasets compared with the ACO/SA. Furthermore, the results obtained by the ACO/GA outperform other hybrid classifiers in all benchmark scenarios. This result is due to the enhancement by using the concepts of multi-neighbourhood structure in GA (i.e., crossover and mutation) to overcome the local optima and find the best classification rules from the dataset.

Table 6 and Fig. 5 and 6 show the results of the nonparametric Friedman test with Holm's post-hoc test in the second scenario. This scenario determines the average classification accuracy rank, average number of discovered rule rank and average model size rank of the statistical results, which are reported in Table 6, across the 12 datasets. Figure 5 displays the results of the average classification accuracy rank versus the average number of rule rank. Figure 6 presents the results of the average classification accuracy rank versus the average model size rank. In all cases, the lowest rank indicates a good algorithm performance.

Table 3: Classification accuracy result (average+/−standard deviation) obtained using 10-fold-cross-validation for all the classifiers

Dataset	ACO/PSO 2	ACO/SA	ACO/GA
Scale Balance	68.66+/-4.97	71.04+/-3.91	**71.22%+/-2.31%**
Ljubljana Breast Cancer	70.94+/-5.37	72.39+/-9.09	**73.06%+/-2.01%**
Wisconsin Breast Cancer	93.86+/-4.56	**96.14+/-2.93**	95.57%+/-0.84%
CreditA	84.69+/-4.39	85.80+/-2.58	**86.52%+/-1.22%**
Diabetes Data	76.31+/-4.32	76.70+/-4.11	**77.72%+/-1.48%**
Cleveland Heart disease	78.51+/-6.16	**81.78+/-7.29**	81.34%+/-2.1%
Statlog Heart disease	78.89+/-7.78	81.11+/-9.14	**81.48%+/-1.56%**
Iris dataset	94.0+/-8.14	93.33+/-8.43	**96%+/-1.09%**
Medical Lymphography	77.19+/-12.59	78.29+/-6.88	**80.26%+/-3.03%**
Mushroom Data	**100.0**+/-**0.0**	99.01+/-2.55	98.52%+/-0.14%
Segment Data	82.08+/-4.64	92.42+/-1.60	**92.56%+/-0.67%**
Vehicle Data	60.64+/-5.18	**69.98+/-4.04**	64.64%+/-2.13%

Table 4: Number of rules result (average+/−standard deviation) obtained using 10-fold-cross-validation methods for all the classifiers

Dataset	ACO/PSO 2	ACO/SA	ACO/GA
Scale Balance	22+/-0	19.20+/-1.72	**8.9+/-0.03**
Ljubljana Breast Cancer	11.3+/-2.05	16.40+/-1.02	**9.2+/-0.42**
Wisconsin Breast Cancer	9.9+/-1.37	11.90+/-0.83	**8.5+/-0.17**
CreditA	20.1+/-1.37	20.40+/-2.33	**13.8+/-0.66**
Diabetes Data	37.1+/-2.23	29.30+/-1.10	**16.4+/-0.69**
Cleveland Heart disease	10.6+/-1.42	12.80+/-0.87	**9.5+/-0.34**
Statlog Heart disease	9.7+/-1.70	12.50+/-1.12	**8.8+/-0.43**
Iris dataset	4.7+/-0.48	4.70+/-0.46	**4.3+/-0.22**
Medical Lymphography	15.4+/-1.34	**7.90+/-0.83**	8.4+/-0.27
Mushroom Data	17.6+/-1.42	24.90+/-1.76	**7+/-0.11**
Segment Data	33.2+/-4.36	57.60+/-2.42	**27.3+/-0.4**
Vehicle Data	30.5+/-3.5	41.30+/-1.35	**23.7+/-0.92**

Table 5: Model size result (average+/−standard deviation) obtained using 10-fold-cross-validation method for all classifiers

Dataset	ACO/PSO 2	ACO/SA	ACO/GA
Scale Balance	52+/-0	42.90+/-5.15	**13.4+/-0.16**
Ljubljana Breast Cancer	26.8+/-6.196	33.20+/-3.74	**16.1+/-0.81**
Wisconsin Breast Cancer	17.1+/-2.42	18.90+/-2.02	**10+/-0.52**
CreditA	70.6+/-7.6	53.50+/-8.88	**26.5+/-2.11**
Diabetes Data	112.5+/-9.312	65.70+/-3.90	**27.8+/-1.79**
Cleveland Heart disease	28.3+/-4.347	29.10+/-3.53	**20.5+/-1.34**
Statlog Heart disease	25.9+/-4.30	27.60+/-3.98	**18.4+/-1.59**
Iris dataset	**3.3**+/-**0.94**	4.80+/-1.08	3.8+/-0.44
Medical Lymphography	42.8+/-6.48	**16.50+/-2.97**	17.7+/-0.87
Mushroom Data	33.4+/-2.87	37.00+/-2.90	**7.1+/-0.1**
Segment Data	59.3+/-7.9	121.60+/-5.97	**42.9+/-1.25**
Vehicle Data	98.2-11.85	116.80+/-7.14	**60.2+/-2.84**

Table 6: Results of the non-parametric Friedman test with Holm post-hoc test based on average performance rank on all datasets

	ACO/PSO 2	ACO/SA	ACO/GA
Accuracy	2.75	1.83	1.42
Rule	2.29	2.63	1.08
Terms	2.25	2.58	1.17

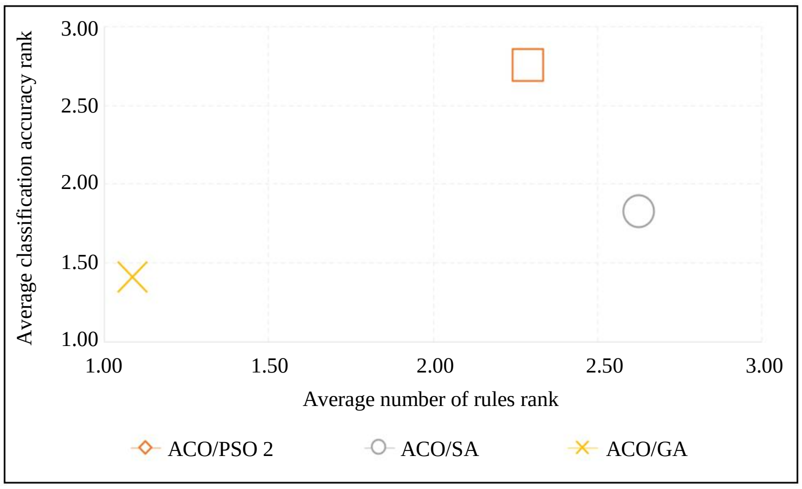

Fig. 5: Results of ACO/GA on the average classification accuracy rank versus the average number of discovered rule rank

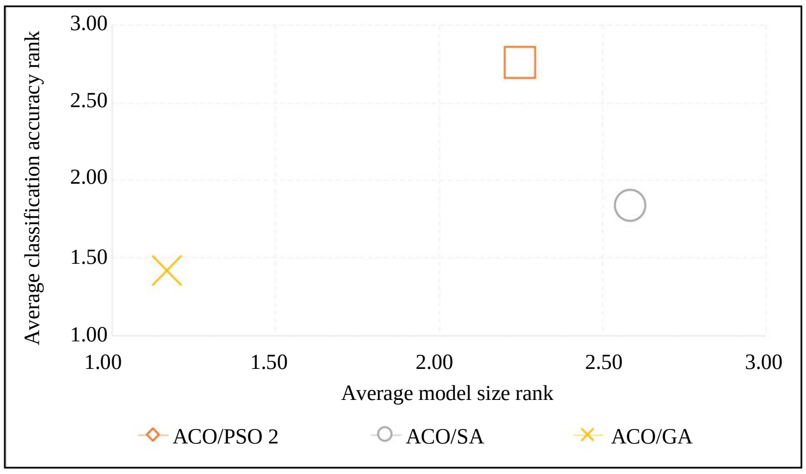

Fig. 6: Results of ACO/GA on the average classification accuracy rank versus the average model size rank

Figure 5 and 6 show that the results obtained by our proposed ACO/GA classifier dominate the other two classifiers. Therefore, the ACO/GA dominates the hybridisation with Ant-Miner classifier in all evaluation criteria. This result is due to the enhancement process achieved by the GA algorithm to the classification rule discovered by the Ant-Miner classifier. GA uses the multiple neighbourhood structures (i.e., crossover and mutation) procedures to escape from local minima.

Conclusion

This study proposes a hybrid Ant-Miner classification algorithm with GA (ACO/GA) classifier. The proposed classifier improves the classification accuracy, the number of discovered rules and classification complexity. The intensification of GA is used to improve the local exploitation of Ant-Miner search. GA uses three components, that is, (i) rule mutation to create a new starting rule from the best-found rules; (ii) crossover to find the best improvement in neighborhood structures; and (iii) acceptance criterion to accept or reject the current accuracy improvement of the generated rule. This type of local search assists the Ant-Miner to enhance the exploitation by focusing on promising areas of the search space. The classification performance of the proposed classifier is tested against other hybridization ant-mining classifiers (i.e., ACO/SA and ACO/PSO2) by using 10-fold cross validation on 12 datasets from UCI. The results of the proposed classifier outperform the other classifiers in all evaluation criteria. In future work, the application of this classifier can be observed by real-world applications in various domains, such as DNA sequence classification, medical diagnosis, credit scoring and text mining. These real-life applications can provide extensive knowledge in the behavior and performance of the proposed classifier. Another research direction is that other stochastic local search algorithms (e.g., randomized iterative improvement, iterated greedy and evolutionary algorithms) can be hybridised with Ant-Miner algorithms.

Funding Information

The researchers would like to thank the Malaysian Ministry of Higher Education for financially supporting this research under Grant Scheme (TRGS/1/2018/UUM/02/3/3 (S/O code14163).

Author's Contributions

All researchers are contributed in this article equally.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and there are no ethical issues involved.

References

Aarts, E. H. L., Korst, J. H. M., & Laarhoven, P. J. M. Van. (1997). Simulated annealing. Local Search in Combinatorial Optimization, 27(3), 797–802.

AL-Behadili, H. N. K. (2018). Intelligent hypothermia care system using ant colony optimization for rules prediction. Journal of University of Babylon for Pure and Applied Sciences, 26(2), 47-56.

Al-Behadili, H. N. K., Ku-Mahamud, K. R., & Sagban, R. (2019). Annealing strategy for an enhance rule pruning technique in ACO-based rule classification. Indones. J. Electr. Eng. Comput. Sci, 16(3), 1499-1507.

Al-Behadili, H. N. K., Ku-Mahamud, K. R., & Sagban, R. A. F. I. D. (2018a). Ant colony optimization algorithm for rule-based classification: Issues and potential solutions. J. Theor. Appl. Inf. Technol, 96(21), 7139-7150.

Al-Behadili, H. N. K., Ku-Mahamud, K. R., & Sagban, R. (2018b, April). Rule pruning techniques in the ant-miner classification algorithm and its variants: A review. In 2018 IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE) (pp. 78-84). IEEE.

Al-Behadili, H. N. K., Ku-Mahamud, K. R., & Sagban, R. A. F. I. D. (2020a). HYBRID ANT COLONY OPTIMIZATION AND ITERATED LOCAL SEARCH FOR RULES-BASED CLASSIFICATION. J. Theor. Appl. Inf. Technol, 98(04), 657-671.

Al-Behadili, H. N. K., Sagban, R., & Ku-Mahamud, K. R. (2020b). Adaptive Parameter Control Strategy for Ant-Miner Classification Algorithm. Indonesian Journal of Electrical Engineering and Informatics (IJEEI), 8(1), 149-162.

Blum, C. (2005). Beam-ACO—Hybridizing ant colony optimization with beam search: An application to open shop scheduling. Computers & Operations Research, 32(6), 1565-1591.

Blum, C. (2008). Beam-ACO for simple assembly line balancing. INFORMS Journal on Computing, 20(4), 618-627.

Blum, C., Vallès, M. Y., & Blesa, M. J. (2008). An ant colony optimization algorithm for DNA sequencing by hybridization. Computers & Operations Research, 35(11), 3620-3635.

Di Caro, G., & Dorigo, M. (1998). AntNet: Distributed stigmergetic control for communications networks. Journal of Artificial Intelligence Research, 9, 317-365.

Dorigo, M., & Stützle, T. (2004). Ant Colony Optimization. Cambridge, MA, USA: MIT Press.

Dorigo, M., & Stützle, T. (2010). Ant Colony Optimization: Overview and Recent Advances. In Handbook of Metaheuristics (Vol. 146, pp. 227–264). Springer US.

Dua, D., & Karra, T. (2017). UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

Holden, N., & Freitas, A. A. (2008). A hybrid PSO/ACO algorithm for discovering classification rules in data mining. Journal of Artificial evolution and Applications, 2008.

Jabbar, A. M., Ku-Mahamud, K. R., & Sagban, R. (2018, April). Ant-based sorting and ACO-based clustering approaches: A review. In 2018 IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE) (pp. 217-223). IEEE.

Jabbar, A. M., Ku-Mahamud, K. R., & Sagban, R. (2019a). Modified ACS Centroid Memory for Data Clustering. J. Comput. Sci, 15(10), 1439-1449.

Jabbar, A. M., Sagban, R. A. F. I. D., & Ku-Mahamud, K. R. (2019b). BALANCING EXPLORATION AND EXPLOITATION IN ACS ALGORITHMS FOR DATA CLUSTERING. J. Theor. Appl. Inf. Technol, 97(16), 4320-4333.

Jabbar, A. M., Ku-Mahamud, K. R., & Sagban, R. (2020). An improved ACS algorithm for data clustering. Indonesian Journal of Electrical Engineering and Computer Science, 17(3), 1506-1515.

Kanan, H. R., Faez, K., & Taheri, S. M. (2007, July). Feature selection using ant colony optimization (ACO): a new method and comparative study in the application of face recognition system. In Industrial Conference on Data Mining (pp. 63-76). Springer, Berlin, Heidelberg.

López-Ibáñez, M., Stützle, T., & Dorigo, M. (2016). Ant colony optimization: A component-wise overview. Handbook of heuristics, 1-37.

Martens, D., Baesens, B., & Fawcett, T. (2011). Editorial survey: swarm intelligence for data mining. Machine Learning, 82(1), 1-42.

Parpinelli, R. S., Lopes, H. S., & Freitas, A. A. (2002). Data mining with an ant colony optimization algorithm. IEEE transactions on evolutionary computation, 6(4), 321-332.

Raymer, M. L., Punch, W. F., Goodman, E. D., Kuhn, L. A., & Jain, A. K. (2000). Dimensionality reduction using genetic algorithms. IEEE transactions on evolutionary computation, 4(2), 164-171.

Ripon, S. H. (2019, February). Rule induction and prediction of chronic kidney disease using boosting classifiers, Ant-Miner and J48 Decision Tree. In 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE) (pp. 1-6). IEEE.

Robu, R., Vaçar, C., Robu, N., & Holban, Ş. (2015, July). A study on Ant Miner parameters. In 2015 6th International Conference on Information, Intelligence, Systems and Applications (IISA) (pp. 1-11). IEEE.

Sagban, R., Ku-Mahamud, K. R., & Abu Bakar, M. S. (2016). Reactive max-min ant system with recursive local search and its application to TSP and QAP. Intelligent Automation & Soft Computing, 23(1), 127-134.

Sagban, R., Ku-Mahamud, K. R., & Bakar, M. S. A. (2017, May). Unified strategy for intensification and diversification balance in ACO metaheuristic. In 2017 8th International Conference on Information Technology (ICIT) (pp. 139-143). IEEE.

Saian, R., & Ku-Mahamud, K. R. (2012, March). Ant colony optimization for rule induction with simulated annealing for terms selection. In 2012 UKSim 14th International Conference on Computer Modelling and Simulation (pp. 33-38). IEEE.