IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 3, Ver. II (May-Jun. 2014), PP 63-72
www.iosrjournals.org

Enhanced cAntMiner$_{PB}$ Algorithm for Induction of Classification Rules using Ant Colony Approach

Safeya Rajpiplawala[1], Dheeraj Kumar Singh[2]
[1]*(Computer Science and Engineering, Parul Institute of Engineering and Technology/ Gujarat Technological University, India)*
[2]*(Information Technology, Parul Institute of Engineering and Technology/ Gujarat Technological University, India)*

Abstract *: Mining classification rules from data is a key mission of data mining and is getting great attention in recent years. Rule induction is a method used in data mining where the desired output is a set of Rules or Statements that characterize the data. Within Rule Induction model, Swarm Intelligence (SI) is a technique where rules may be discovered through the study of joint behavior in decentralized, self-organized systems, such as ants. Ant-Miner is a rule induction algorithm that uses SI techniques to form rules. The main idea of this study is to discover the suitability of ant colony optimization for constructing accurate classifiers which can be learned in practical time even for big datasets. The cAntMiner$_{PB}$ algorithm is an extension of the cAntMiner algorithm. The main task is to modify the existing algorithm cAntMiner$_{PB}$ to allow each rule to dynamically select rule quality evaluation function and to improve the accuracy and preserving rule list simplicity. In this study, we examine the use of different rule quality evaluation functions for rule quality assessment prior to pheromone update and check how the use of different evaluation function affects the output model in terms of predictive accuracy and model size. In experimental results, we use 10 different rule quality evaluation functions on 12 benchmark datasets and found that predictive accuracy obtained by new proposed method is statistically significantly higher than the predictive accuracy of existing algorithm.*

Keywords: *ACO, SI, Rule Classifier, AntMiner*

I. INTRODUCTION

There is an extensive variety of algorithms that have come out from the actions of social bugs. Social insects are typically differentiated by their self union and with the negligible contact or the absence of it. Every social insect separately is superior. They can get information regarding surroundings and cooperate with the remote insects in some way, by stigmergy. All these features describe Swarm Intelligence. In section 2 details are described of ACO and AntMiner Algorithm, in section 3 cAntMiner Algorithm, in section 4 cAntMiner$_{PB}$ Algorithm, in section 5 proposed work, in section 6 implementation tools and in section 7 Experimental results are shown on 12 datasets taken from UCI Machine Learning Repository. Finally in section 8 Applications and in Section 9 Conclusion is described.

II. ACO AND ANTMINER ALGORITHM

Ant colony optimization (ACO) [1] is a stem of a recently developed form of artificial intelligence known as swarm intelligence. Swarm intelligence (SI) is "the property of a structure whereby the combined activities of (simple) agents work together locally with their surroundings cause simple functions overall patterns to appear" [2]. In cluster of insects, which exist in colonies, such as ants a creature cannot do a work on its own; colony's supportive work is the key motivation shaping the intelligent behavior. The majority of actual ants are blind. Real ants whilst walking arbitrarily leaves a chemical substance known as pheromone [1]. Pheromone magnetizes other ants to stay close to other ants. The pheromone vanishes over time to allow search evaporation. In many experiments presented by Dorigo and Maniezzeo give details about the complex behavior of colonies [3], where ants always prefer shortest path.

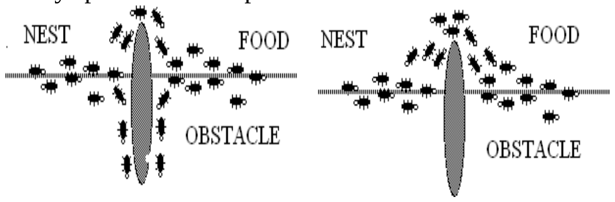


Figure1. Overview of Dorigo and Maniezzeo experiment [3]

Further Parpinelli, Lopes and his associates were the first to propose Ant Colony Optimization (ACO) for learning classification rules, called Ant-Miner. They discover that an ant-based search is more flexible, robust

and optimized than conventional approaches. Their technique employs a heuristic value based on entropy measure. The goal of Ant Miner is to mine classification rules from data [5]. It follows a sequential covering approach to learn a list of classification rules from the given data set. Then rules are included to the list of discovered rules. Then the training cases covered correctly by these rules, are subtracted from the training set. It covers all training cases. Now classification rule is of type:
IF < t1 AND t2 AND…> Then < C >.Where t1=term1, t2=term2, C=Class

A. Pheromone Initialization
The initial amount of pheromone dropped at each path is inversely proportional to the number of values of all attributes, and is defined by

$$\mathrm{Tij}(t=0) = 1/\sum_{i=1}^{a} \mathrm{bi} \qquad (1)$$

Where 'a' is the total number of attributes, 'bi' is the number of values in the area of attribute i.

B. Rule creation
Each rule in Ant miner includes a condition part as the ancestor and an expected class. The condition part is a combination of attribute-operator-value tuples. Here we assume rule condition such as termij Bi =Wij, where Bi is the ith attribute and Wij is the jth value in the area of Bi. The likelihood that this clause is included to the current incomplete rule, is given by

$$Pij(t) = \frac{\mathrm{ij(t).ij}}{\sum_{1}^{a}\mathrm{xi}\ \sum_{j=1}^{bi}(\mathrm{ij(t).\ ij})} \qquad (2)$$

Where Nij is a problem dependent heuristic value for termij. Tij is the amount of pheromone at present available (at time t) on the link between attribute i.

C. Heuristic Value
In Ant Miner, the heuristic value is an information theoretic measure for the excellence of the term to be added to the rule which is measured in terms of the entropy for choosing this term to the others, and is given by the following equations:

$$\eta\mathrm{ij} = \frac{\log_2 \mathrm{k} - \mathrm{Info\ Tij}}{\sum_{i=1}^{a} xi \sum_{j=1}^{bi}(\log_2 k - \mathrm{Info\ Tij})} \qquad (3)$$

$$\mathrm{Info\ Tij} = \sum_{w=1}^{k}\left[\frac{\mathrm{Freq\ Tij}^{W}}{|\mathrm{Tij}|}\right] * \log_2\left[\frac{\mathrm{Freq\ Tij}^{W}}{|\mathrm{Tij}|}\right] \qquad (4)$$

Where k is the number of classes, |Tij| is the total number of cases in partition Tij (partition containing cases where attribute Ai has value Vij), FreqTij^{w} is the number of cases in partition Tij with class w. The high the value of info Tij, the less expected that the ant will prefer termij.

D. Rule Pruning
The rule pruning procedure iteratively removes the term whose elimination will lead the maximum increase in the quality of the rule. The quality of rule is measured using the following equation:

$$\mathrm{Q} = \frac{\mathrm{TP}}{\mathrm{TP+FN}} \cdot \frac{\mathrm{TN}}{\mathrm{FP+TN}} \qquad (5)$$

Where, *TP* is the numeral of cases covered by the rule and having the similar class that is expected by the rule, *FP* is the numeral of cases covered by the rule and having a class that was not expected by the rule, *FN* is the numeral of cases that are not covered by the rule, whilst having the class that is expected by the rule, *TN* is the numeral of cases that are not covered and which have a different class from the class that is expected by the rule.

E. Pheromone changing rule
After Rule construction finishes, pheromone are revised as follows:

$$\mathrm{\tau ij}(t+1) = \tau ij(t) + \tau ij(t).Q \ ; \ \forall i,j \quad R \qquad (6)$$

For replicating the phenomenon of evaporation, the amount of pheromone related with each termij which does not take place in the assembled rule must be reduced. To decrease the pheromone of an unused term, divide the value of each Tij by the summation of all Tij.

Ant Miner Issues and Challenges:
Ant-Miner generates simpler (smaller) rule lists than CN2. It give the impression of being beneficial as it is considerable to reduce the number of discovered rules and rule terms (conditions) in order to increase lucidity of the discovered facts, where it will be exposed to a person as a support for clever decision making.

Guidelines for potential research: First, extend Ant-Miner to cope with continuous attributes, so no requirement of discretization in a preprocessing step. Second, discover the presentation of other kinds of heuristic function and pheromone changing approach.

III. CANTMINER ALGORITHM

Ant miner needs the discretization method as a preprocessing and it is appropriate only for the nominal attributes. The drawback of this approach is that less information will be available to the classifier. Generally real-world classification problems are described by nominal (discrete values) and continuous attributes. Fernando, Freitas, and Johnson proposed an extension to Ant-Miner, named cAntMiner, which was able to handle the continuous values as well [8].

Purpose:

Ant miner that can handle continuous attributes to extract classification rules from data. Entropy based discretization used during rule construction to create Discrete intervals "on-fly".

Method:

Dynamic Discretization of continuous value is done by *Entropy based discretization* method. If nominal attribute, where every *term*$_{ij}$ has the form ($a_i = v_{ij}$), the entropy for the attribute-value pair is computed as in equation (7) – used in the original Ant-Miner:

$$entropy(a_i = v_{ij}) \equiv \sum_{c=1}^{k} -p(c|\, a_i = v_{ij}).\log_2(p(c|a_i = v_{ij})) \qquad (7)$$

where,

$p(c \mid a_i = v_{ij})$ is the experiential probability of examining class c conditional on having observed $a_i = v_{ij}$, k is the number of classes.

For computing the entropy of nodes representing continuous attributes (term$_i$) as these nodes do not represent an attribute-value pair, we need to choose a threshold value v to dynamically partition the continuous attribute a_i into two intervals: a < v and $a_i >= v$.

The best threshold value is the value v that minimizes the entropy of the partition, given by equation (8):

$$ep_v(a_i) \equiv \frac{|S_{ai<v}|}{|S|}.entropy(a_i < v) + \frac{|S_{ai>v}|}{|S|}.entropy(a_i \geq v) \qquad (8)$$

where

$|Sa_i<v|$ is the total number of cases in the partition

$a_i < v$ (partition of training cases where the attribute a_i has a value less than v),

$|Sa_i>=v|$ is the total number of cases in the partition

$a_{i>=}$ v (partition of training cases where the attribute ai has a value greater or equal to v)

|S| is the total number of training cases.

After selecting threshold v_{best}, the entropy of the term matching to the minimum entropy value of the two partitions and it is defined as:

$$entropy(term_i) \equiv \min\,(entropy(a_i < v_{best}), entropy(a_i \geq v_{best})) \quad (9)$$

Guidelines for further research: First, examine the performance of special discretisation methods in the rule construction process. Second, evaluate other kinds of pheromone updating methods as dropping pheromone on the edges tends to a customized pruning process.

IV. CANTMINER$_{PB}$ ALGORITHM

In this [9] a new approach is proposed, which directs the search performed by the algorithm using the quality of a candidate list of rules, instead of a sole rule. The main motivation is to avoid the problem of rule interaction derived from the order in which the rules are discovered i.e., the outcome of a rule affects the rules that can be discovered subsequently since the search space is modified due to the removal of examples covered by previous rules. In the new sequential covering strategy proposed, the pheromone matrix used by the ACO algorithm is extended to include a tour identification that indirectly encodes the sequence in which the rules should be created, allowing a more effective search for the best list of rules.

V. PROPOSED WORK

5.1 Overview of Proposed Work:

In cAntMiner$_{PB}$ algorithm, an ACO based procedure is used to construct a complete list of rules. In this the search is performed and optimized to find the best list of rules. The search is directed by the quality of a candidate list of rules. The algorithm uses a rule quality function to prune (i.e., remove irrelevant terms) from a candidate rule. Therefore, there are two quality functions involved in the search for the best list of rules in this algorithm: the rule quality function used during pruning and the rule-list quality function used to guide the search (i.e., update pheromones).

Improving the rule quality function of sequential covering algorithms tends to improve the overall performance of the algorithm.
In cAntMiner$_{PB}$, ants find path through a fully connected graph of all possible rule terms (attribute-value pairs) in order to construct rules.
Here preliminary approach to dynamically select the rule quality function is based around addition of extra vertices to the construction graph including the candidate rule terms to represent the available rule quality functions, resulting in one large graph. This simple approach had the benefits that it is an easy concept and it fits very nicely into cAntMiner$_{PB}$ with few modifications to the existing algorithm.

5.1.1 Method description (for selecting Rule Quality Evaluation Function dynamically)
The modification of cAntMiner$_{PB}$ algorithm is proposed, to dynamically select the rule quality function to be applied during the pruning procedure (per rule). Here different rules can be pruned using different rule quality functions listed in table 1.
Now the use of two separate graphs is needed:

1. Graph consist of different rule quality functions
2. Graph consists of rule terms

In this proposed scheme when creating a rule, an ant will first visit the Rule Quality Function graph to select an evaluation function, and then visit the rule terms graph to create the rule.
Here to achieve the needed functionality few modifications are done in original algorithm.
Steps for modification:
1. First create a Rule Quality Function graph and initialize its pheromones.
2. Each ant would select the rule quality function before creating a rule (dynamic selection per rule can be achieved)
3. Then store the rule quality for later use in the pruning stage.
4. The selection of the Rule Quality Function is based on pheromone levels, no heuristic information is used.
5. Once the iteration-best list of rules has been generated, the two pheromone matrices would be updated to react to the selected RQFs as well as the list of rule terms used in the list of rules.

5.1.2 Modified cAntMiner$_{PB}$ algorithm:
Input: training examples (Dataset)
Output: best discovered list of Classification rules that will classify data according to the predetermined class

```
1. Create a Rule Quality Function graph and initialize its pheromones
2: Initialize all trails with same amount of pheromone
3: Start with empty list of rules
4: set convergence test index (m)
5: while stopping criteria not met do
6: Set list of iteration best rules to empty
7:for each ant in the colony do
8: consider all training examples
9:        while convergence criteria not satisfied do
10:       Each Ant will choose rule quality function before creating a rule
12:       create rule;
13:       Prune(rule);
14:       Store the rule quality for later use
15:             remove training examples correctly classified by pruning best rule
16:             add the rule in discovered list of rules
17:       end while
18:       if compare quality of discovered list of rules then
19:       update list according to highest quality
20:       end if
21: end for
22:       Update both Pheromones metrics
23:       if compare global list of rules with iterative best list of rules then
24:       update global list accordingly
25:       end if
26:       reinitialize training test
27: end while
28: return list of global best rules
```

5.2 Experimental Parameters

Input Parameters:

1. Total no of ants. (n) *(colony size) :* The maximum number of complete rules created and pruned. In each iteration the best rule found is considered as discovered rule. The larger *no of ants*, the more rules are evaluated per iteration, but the system will become slow.
2. Maximum no of Iterations: The convergence test is able to stop the search before reaching the maximum value.
3.MAX- MIN Evaporation Factor (*ρ*):pheromone evaporation is simulated by decreasing the amount of pheromone of each entry by a user-defined factor *ρ.*
4. Minimum no of examples covered by rule (min _cases)*:* Each rule should cover at least min_cases per rule.
5. Maximum no of uncovered cases in training set *(Max_uncover_cases):* The process of rule finding is continued until the number of training cases that are not covered by any discovered rule is smaller than this threshold.

5.3 Quality Evaluation Functions

The rule evaluation function is the root of sequential covering algorithms. In Ant-Miner (and its variations), each time an ant creates a rule, its quality is calculated and the rule is only considered if it has the best quality in that iteration.
In cAntMiner$_{PB}$, the evaluation function is used during the pruning step, as the search is guided by the quality of a list of rules. Unlike Ant-Miner and its variations, in the modified cAntMiner$_{PB}$ algorithm it uses two quality functions in its search process:
1. Rule quality function: It is used to take decision about pruning an individual rule.
2. List quality function: It guides the search (i.e., the pheromone update is based on the quality of a candidate list of rules).
So here in proposed algorithm the effects of rule quality functions and list quality functions can be checked and based on the results we will check how much effect each has on the proposed algorithm in terms of predictive accuracy and size of the discovered model, and compare the results against the default combination used in original cAntMiner$_{PB}$ algorithm.
In order for dynamic rule quality function selection process we needed a wide selection of different rule quality functions.
We have selected previously used rule quality functions described in [15, 16, 17], as well as the original rule quality function used in cAntMinerPB (Sensitivity X Specificity).
The selected functions are listed in Table 1. For the parametric quality functions, we have used their default parameter values [13] (shown in the `Parameter' column in Table 1).

Table 1: The Rule Quality Functions used in the dynamic selection process.

Function Name	Parameter	Formula
1.Sensitivity×Specificity	--	$TP/(TP+FN) X TN/(TN+FP)$
2. Confidence + Coverage	--	$TP/(TP+FP)+TP/S$
3. Jaccard	--	$TP/(TP+FP+FN)$
4. Klosgen	w=0.4323	[(TP+ FP) / S]^ω. ($TP/(TP+FP)-(TP+FN)/S$)
5. M-Estimate	m=22.466	$(TP+m.(TP/S))/(TP+FP+m)$
6. Accuracy	--	$(TP+FN)/(TP+FP+TN+FN)$
7. Cost Measure	c=0.437	(c . TP) - ((1- c). FP)
8. F-Measure	β= 0.5	$\frac{(1+\beta.\beta)\ .\ TP/(TP+FN).TP/(TP+FP)}{(\beta.\beta)\ .\ TP/(TP+FN)+TP/(TP+FP)}$
9. Relative Cost Measure	cr=0.342	(cr.recall) – ((1-cr). (FP/ (FP+TN))
10. C4.5' Error Based Function	--	U_{CF} (FP; TP + FP)

Where the terms TP, FP, TN, and FN are explained in Section 2 and S The total number of examples (TP + FP + TN +FN)

5.4 Working of Proposed Method:

The work consist of let dynamically selecting the rule quality function to be used in the pruning procedure (per rule fashion), where different rules can be pruned using different rule quality functions.
As has been previously studied [9, 10], rule quality functions have different bias and capture different aspects of the rule (e.g., some might favor consistency over coverage).

5.4.1 Selecting rule evaluation functions per rule

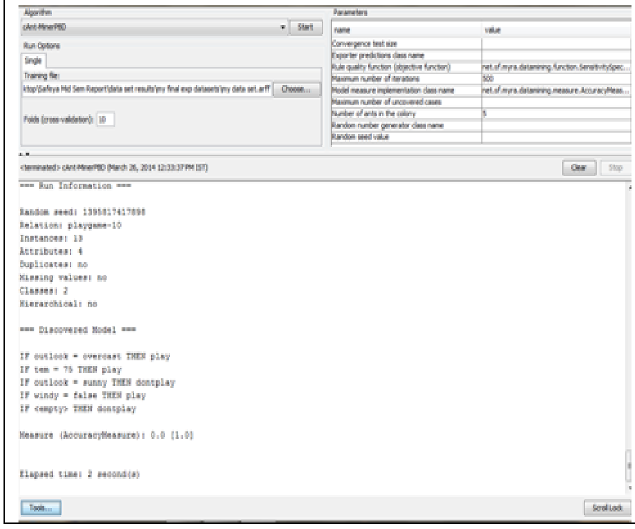
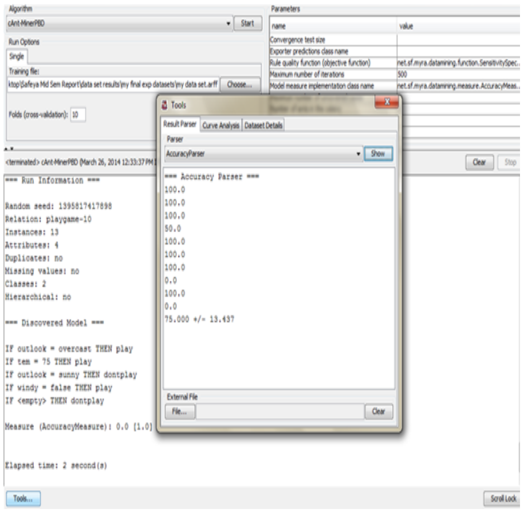
We used two separate construction graphs as explained in section 5.1.1. Here the convergence tests had to solely rely upon the terms selected to create the rules.
As the rule quality functions and rules terms graphs are independent, the pheromones in cAntMiner$_{PB}$ are preserved in sequence. It means the first rule being chosen in a list has a list of pheromones which is saved and updated across iterations; same is with the second rule and so on.
This means that the default rule (with an empty antecedent) does not have a function associated.
The first rule is now converging to the list of terms, which was affected by the choice of the rule quality function, both of which may be very different to the terms and quality function used by the second rule.
Convergence is simply determined by examining the rule terms graph, since different rule quality functions can lead to the same rule and the choice of the rule quality function does not affect the quality of the list of rules (as long as they produce the same rules).
Table2 presents an example of a list of rules with associated rule quality functions

Table 2: An example of a list of rules with associated rule quality functions.

Rules Generated	**Rule Quality Function**
IF petal-width <= 0.8 THEN Iris-setosa	USING F-measure function
IF petal-width > 1.75 THEN Iris-virginica	USING Error-based function
IF sepal-length <= 6.15 THEN Iris-versicolor	USING M-Estimate function
IF <empty> THEN Iris-virginica	Default

VI. IMPLEMENTATION TOOLS

The implementation phase of any system development is the most important phase as it yields the solution, which solves the problem at hand. In implementation stage theoretical design created in the design phase is converted into a working system. It is the most critical stage in development of a new system because it involves study of the existing system, careful planning, constraints on implementation, designing of methods to make changes, and evaluation of newly created system.
Ant Miner is a cross-platform Ant Colony Optimization framework written in Java. It provides a specialized data mining layer to support the application of ACO to classification problems, including the implementation of Ant-Miner and cAntMiner algorithms.

Main Window	**Tools Window**
	

Algorithm: cAnt-MinerPBD
Run Options: Single
Training file: ktop\Safeya Mid Sem Report\data set results\my final exp datasets\my data set.arff
Folds (cross-validation): 10

name	value
Convergence test size	
Exporter predictions class name	
Rule quality function (objective function)	net.sf.myra.datamining.function.SensitivitySpec...
Maximum number of iterations	500
Model measure implementation class name	net.sf.myra.datamining.measure.AccuracyMeas...
Maximum number of uncovered cases	
Number of ants in the colony	5
Random number generator class name	
Random seed value	

<terminated> cAnt-MinerPBD (March 26, 2014 12:33:37 PM IST)

```
=== Run Information ===

Random seed: 1395817417898
Relation: playgame-10
Instances: 13
Attributes: 4
Duplicates: no
Missing values: no
Classes: 2
Hierarchical: no

=== Discovered Model ===

IF outlook = overcast THEN play
IF tem = 75 THEN play
IF outlook = sunny THEN dontplay
IF windy = false THEN play
IF <empty> THEN dontplay

Measure (AccuracyMeasure): 0.0 [1.0]


Elapsed time: 2 second(s)
```

Tools — Result Parser — Parser: AccuracyParser

```
=== Accuracy Parser ===
100.0
100.0
100.0
50.0
100.0
100.0
100.0
0.0
100.0
0.0
75.000 +/- 13.437
```

VII. EXPERIMENTAL RESULTS

7.1 Performance Metrics

Our performance metrics are: predictive accuracy, number of rules, and the number of terms per rule.

1. Predictive Accuracy

The predictive accuracy is defined as the percentage of true predictions among all predictions on the test set. The experiments are performed using a ten-fold cross validation procedure. A dataset is divided into ten equally sized, mutually exclusive subsets. Each of the subset is used once for testing while the other nine are used for training. The results of the ten runs are then averaged and this average is reported as final result.

2. Number of Rules
This is the average of number of rules in the rule sets obtained by ten-fold cross validation.
3. Number of Terms per Rule
This is the average of terms per rule in all the rule sets obtained by ten-fold cross validation.

7.2 Parameters Settings

The cAntMiner$_{PB}$ has five user defined parameters: number of ants, maximum uncovered cases, evaporation factor, minimum covered examples per rule value, maximum number of iterations. The values of these parameters are given in Table 3. These values have been chosen because they seem reasonable and have been used by [9].

Table 3: Parameters used in experiment

PARAMETER NAME	VALUE
Minimum covered examples per rule value	10
Maximum number of iterations	500
Maximum number of uncovered cases	1.0
Number of ants in the colony	5
Evaporation factor	0.90
Cross Validation Fold	10

7.3 Datasets:

For the experiments, we have used following datasets obtained from the UCI repository [14] and one sample weather dataset. The main characteristics of the datasets are summarized in Table 4.

Table 4: Characteristics of datasets used in cAntMiner$_{PB}$

Data Sets	Attributes		Classes	Size
	Nominal	Continuous		
Breast-w	0	30	2	569
Credit-a	8	6	2	690
Dermatology	33	1	6	366
Glass	0	9	7	214
Ionosphere	0	34	2	351
Iris	0	4	3	150
Liver-disorders	0	6	2	345
Parkinsons	0	22	2	195
Wine	0	13	3	178
Zoo	16	0	7	101
Hepatitis	13	6	2	155
Weather	4	0	2	14

7.4 Results of Proposed cAntMiner$_{PBD}$ Algorithm:

We obtained and compared the results of our algorithm with those for existing cAntMinerPB, and proposed cAntMinerPBD and cAntMinerPBD2 (with two functions named jaccard and sensitivity x specificity functions).
The parameter of comparison are predictive accuracies, average number of rules per discovered rule set, and average number of terms per rule and are shown in Table 6, Table 7 and 8. All results are obtained using ten-fold cross validation.
For easy representation of experimental results, in tables' method1 and Method2 names are used as a column title.
Here Method1, Method2 are compared by keeping benchmark as Existing cAntMinerPB algorithm.
Method 1: cAntMinerPBD (Dynamic Selection with all Rule Quality Functions used)
Method 2: cAntMinerPBD2 (Dynamic Selection with two functions jaccard and sensitivity X Specificity)
Resulting list of rules on Weather Dataset for Proposed cAntMiner$_{PBD}$ Algorithm is shown in following table.

Table 5: Resulting rule list on weather dataset

Relation: Weather
Instances: 14
Attributes: 4
Duplicates: no
Missing values: no
Classes: 2
	IF outlook = overcast THEN play
	IF humidity = 70 THEN play
	IF outlook = sunny THEN dontplay
	IF windy = false THEN play
	IF <empty> THEN dontplay
	
	Measure (AccuracyMeasure): 0.5 [1.0]

Table 6: Average Predictive Accuracy (*average* ± *standard error*) in %, measured by 10-fold Cross validation

Average Predictive Accuracy			
Dataset	**Existing algorithm (cAntMiner$_{PB}$)**	**Method 1 (cAntMiner$_{PBD}$)**	**Method 2 (cAntMiner$_{PBD2}$)**
Breast-w	94.729 +/- 0.655	94.137 +/- 1.168	**92.979 +/- 1.124**
Credit-a	**86.951 +/- 1.514**	81.636 +/- 1.191	84.222 +/- 1.193
Dermatology	89.962 +/- 1.924	93.920 +/- 1.195	**93.920 +/- 1.969**
Glass	**70.895 +/- 3.349**	67.763 +/- 3.130	70.895 +/- 2.472
Ionosphere	**91.704 +/- 1.303**	87.964 +/- 2.876	89.546 +/- 1.232
Iris	94.835 +/- 1.930	94.780 +/- 2.251	**96.319 +/- 1.229**
Liver-disorders	**68.065 +/- 2.227**	61.290 +/- 3.003	66.452 +/- 3.049
Parkinsons	**88.039 +/- 2.319**	83.987 +/- 2.050	81.144 +/- 2.057
Wine	92.500 +/- 1.816	95.000 +/- 1.250	**96.250 +/- 1.909**
Zoo	87.091 +/- 2.617	93.091 +/- 2.122	**94.091 +/- 1.611**
Hepatitis	78.750 +/- 2.849	83.750 +/- 3.715	**84.042 +/- 3.378**
Weather	70.000 +/- 11.055	65.000 +/- 13.017	**80.000 +/- 11.055**

Table 7 Average number of terms per rule Details

Average No. of Terms per Rule			
Dataset	**Existing algorithm (cAntMiner$_{PB}$)**	**Method 1 (cAntMiner$_{PBD}$)**	**Method 2 (cAntMiner$_{PBD2}$)**
Breast-w	**1.027 +/- 0.035**	1.454 +/- 0.044	1.142 +/- 0.059
Credit-a	**1.170 +/- 0.070**	1.907 +/- 0.105	1.107 +/- 0.035
Dermatology	2.455 +/- 0.083	**1.681 +/- 0.054**	1.700 +/- 0.073
Glass	**1.246 +/- 0.045**	2.024 +/- 0.052	1.599 +/- 0.075
Ionosphere	**1.090 +/- 0.042**	1.466 +/- 0.053	1.278 +/- 0.058
Iris	**0.937 +/- 0.056**	1.130 +/- 0.048	0.990 +/- 0.037
Liver-disorders	**1.146 +/- 0.050**	2.386 +/- 0.024	1.149 +/- 0.031
Parkinsons	**0.940 +/- 0.035**	1.190 +/- 0.047	0.999 +/- 0.018
Wine	**0.760 +/- 0.007**	1.201 +/- 0.058	1.005 +/- 0.028
Zoo	**1.150 +/- 0.039**	1.642 +/- 0.069	1.852 +/- 0.109
Hepatitis	1.359 +/- 0.076	1.463 +/- 0.068	**1.068 +/- 0.035**
Weather	0.000 +/- 0.000	**0.955 +/- 0.030**	0.985 +/- 0.050

Table 8 Average number of rules Details

Average No. of Rules			
Dataset	**Existing algorithm (cAntMiner$_{PB}$)**	**Method 1 (cAntMiner$_{PBD}$)**	**Method 2 (cAntMiner$_{PBD2}$)**
Breast-w	**7.100 +/- 0.180**	15.300 +/- 0.790	10.300 +/- 0.473
Credit-a	**10.300 +/- 0.335**	35.700 +/- 1.001	23.300 +/- 1.001
Dermatology	**17.200 +/- 0.291**	33.800 +/- 0.663	29.200 +/- 0.663
Glass	**8.800 +/- 0.327**	26.900 +/- 0.752	18.300 +/- 0.746
Ionosphere	**8.200 +/- 0.291**	18.200 +/- 0.467	13.800 +/- 0.629
Iris	**4.400 +/- 0.221**	8.600 +/- 0.371	6.900 +/- 0.348
Liver-disorders	**9.600 +/- 0.371**	41.500 +/- 1.118	18.800 +/- 0.680
Parkinsons	**6.200 +/- 0.200**	19.100 +/- 1.048	16.100 +/- 0.823
Wine	**4.200 +/- 0.133**	8.200 +/- 0.467	6.300 +/- 0.448
Zoo	**6.000 +/- 0.000**	11.500 +/- 0.719	11.000 +/- 0.365
Hepatitis	**6.800 +/- 0.200**	19.200 +/- 0.867	14.900 +/- 0.458
Weather	**1.000 +/- 0.000**	4.900 +/- 0.180	5.200 +/- 0.200

The results indicate that the cAntMinerPBD2 achieves higher accuracy rate than the two compared algorithms for all the datasets. However, the number of rules and the number of terms per rule generated by our proposed technique is mostly higher. The reason is that we allow the generation of rules with low coverage. It means those rules which cover only a few training examples are also allowed.

VIII. Applications

A predominantly successful research direction in ant algorithms is devoted to their use to discrete optimization problems. The early applications of ACO were in the area of NP-hard combinatorial optimization problems. Ant colony optimization has been used effectively to a large number of complex combinatorial problems such as the traveling salesman problem, the quadratic assignment problem, scheduling problems and routing in telecommunication networks. Above all triumphant example of ACO algorithm in this area is Ant Net. Another instance of interesting research direction is swarm robotics, where the aim is to apply swarm intelligence techniques to manage large groups of cooperating sovereign robots. It can be utilized for solving stochastic, dynamic, multi objective, mixed variable optimization problems.

IX. Conclusion

Data is a very important and expensive asset. Data mining is an active area of research. It predicts future trends and behaviors, which helps to make practical, knowledge-driven decisions. Without applying automatic data mining methods it is difficult to successfully examine huge amounts of data. One of the data mining tasks is the classification rules extraction from databases helps in achieving this. Researchers are interested in finding efficient and accurate classification models that achieve higher accuracy rate, are comprehensible and can be learnt in practical time, even for large databases. Ant Miner is a technique that successfully incorporates swarm intelligence in data mining. The primary goal of this work is to develop accurate, comprehensible and efficient classification algorithm based on ACO.

The main objective of research is related to improve accuracy and generate better classification rules. Towards this end a modified algorithm cAntMiner$_{PB}$ is proposed that dynamically selects rule quality evaluation functions.

We have found that by this method, the same rules can be created by the pruning procedure but with different rule quality functions.

This affected the convergence of the algorithm, since the choices of the rule quality functions were not unique and, as a result, there was no selective force towards a particular rule quality function. This meant that the algorithm would rarely converge.

This algorithm can deal with both nominal and continuous attributes, unlike the original Ant-Miner algorithm, which can handle nominal attributes only. The main feature of the proposed method is that there is no

predefined number of rules required to create a candidate list of rules. Also ants have the flexibility of creating lists of different lengths.
More experiments can be carried out on the proposed algorithm on most of the datasets and compare predictive accuracies with other algorithms commonly used for such purpose.

References

[1] DorigoM.,&Caro,G.D,"Ant Algorithm for Optimatisation" , Artificial Life,5(3),137 ,1999.

[2] Bonabeau, E., Dorigo,M., & Theraulaz, G."Swarm Intelligence: From Natural to Artifical System", New York: Oxford University Press, 1999.

[3] Dorigo,M.,&Maniezzo,V."The ant System: Optimization by a colony of cooperating Agents", IEEE Transactions on Systems, Man, and Cybernectics,26(1),1-13,1996.

[4] M. Dorigo and T. Stuetzle.,"Ant Colony Optimization", MIT Press, 2004.

[5] J.Kennedy and R.Mendes,"Population structure and Particle Swarm Performance", Procedeing of IEEE Conference on Evolutinary Computation, Honolulu, Hawalii USA, 2002.

[6] R.S.Parnelli, H.S.Lopes and A.A.Freitas, "Data Mining with an Ant Colony Optimization Algorithm", IEEE Trans. On Evolutionary Computation, special issue on Ant colony Algorithm, 6(4), 321-332,2002.

[7] J.Kennedy,R.C.Eberhart,Y.Shi, "Swarm Intelligence", san Franciso: Morgan Kaufmann/Academic Press 2001.

[8] Fernando E.B. Otero, Alex A. Freitas, and Colin G.Johnson, "cAntMiner: An Ant Colony Classification Algorithm to Cope with Continuous Attributes", Springer-Verlag Berlin, Heidelberg 2008.

[9] F. Otero, A. Freitas, and C. Johnson. "A new sequential covering strategy for inducing classification rules with ant colony algorithms", To appear in IEEE Transactions on Evolutionary Computation, 2012.

[10] Matthew Medland, Fernando E. B. Otero," A Study of Different Quality Evaluation Functions in the cAntMinerPB Classification Algorithm", Philadelphia, Pennsylvania, USA, GECCO' 2012.

[11] D. Martens, B. Baesens, and T. Fawcett, "Editorial survey: Swarm intelligence for data mining," Mach. Learn., vol. 82, no. 1, pp. 1–42, 2011.

[12] Dorigo, M., Stützle, T.: Ant Colony Optimization. MIT Press. 2004

[13] Janssen, F., Furnkranz, J.: On the quest for optimal rule learning heuristics. Machine Learning, 2010.

[14] S. Hettich, and S.D. Bay, "The UCI KDD Archive". Irvine, CA: Dept. Inf. Comput. Sci., Univ. California, 1996 [Online]. Available: http:// kdd.ics.uci.edu

[15] Fayyad, U., G. Piatetsky-Shapiro and P. Smyth,: "The KDD process for extracting useful knowledge from volumes of data". Communications of the ACM, 1996.

[16] Jaiwei, H., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann, 2006.

[17] Dorigo, M., Gambardella, L. M. Ant Colony System: A cooperative learning approach to the travelling salesperson problem. IEEE Transactional on Evolutionary Computation, vol. 1, pp. 53 – 66, 1996.

[18] Dorigo, M.: "Optimization Learning and Natural Algorithms, Ph.D.thesis (in Italian)". Dipartimento di Elettronica, Politecnico di Milano, 1992