International Journal of Innovative Computing, Information and Control
Volume **7**, Number **11**, November **2011**

 ISSN 1349-4198
pp. 6147–6159

ACO BASED DISCOVERY OF COMPREHENSIBLE AND ACCURATE RULES FROM MEDICAL DATASETS

Abdul Rauf Baig, Waseem Shahzad, Salabat Khan and Fariha Altaf

Department of Computer Science
National University of Computer and Emerging Sciences
15-Mauve Area, Rohtas Road, Sector H-11/4, Islamabad, Pakistan
rauf.baig@nu.edu.pk

Received June 2010; revised October 2010

Abstract. *In many real world applications, comprehensibility of the classifier is as important as its accuracy. The medical field is one of those where this requirement is more pronounced. It is not enough for users in this field to have an accurate classifier, and they also need to verify and analyze the logic of the classification process. It is difficult to have confidence in a black box type of classifier when the classification decision is a matter of life and death of a patient. In recent years, algorithms for classification rule discovery based on the ant colony optimization meta-heuristic (ACO) have been proposed, which fulfill both the requirements of high accuracy and comprehensibility. This paper reports some improvements in a recently proposed ACO based classification algorithm, called CAntMiner, whose main feature is a heuristic function based on the compatibility of pairs of attribute-values and class labels, and its application on medical datasets. We study the performance of the algorithm for twelve commonly used datasets and compare it with ten well known classification algorithms, three of which are ACO based. Experimental results show that the accuracy rate obtained by CAntMiner is better than that of the compared algorithms. We also discuss some other issues related to comprehensibility of the classifier building process.*

Keywords: Ant colony optimization (ACO), Classification rules, Medical data mining, Swarm intelligence

1. **Introduction.** Vast quantities of medical data are added to clinical databases on a daily basis. One of the major challenges in this context is to automatically extract accurate and comprehensible knowledge from these large collections of raw data. Discovery of relationships and patterns within these data has the potential to provide new medical knowledge. The discovered knowledge can then be used to improve the quality of service. For example, it can be used by medical practitioners to avoid drugs with adverse effects, suggest less expensive but therapeutically equivalent alternatives, avoid redundant tests and replace expensive tests by cheaper tests. If the extracted knowledge is comprehensible, domain experts can further refine and enhance the discovered knowledge.

Some methods and algorithms already exist for the analysis of medical data. For example, in recent years, neural networks have been extensively applied to many problems [1] including medical related tasks [2-4]. Most applications of neural networks in medicine are related to classification tasks [2]. However, neural networks are incomprehensible and opaque to humans. In most of the medical applications, both comprehensibility and accuracy are required.

A well known example of a comprehensible classifier is the decision tree which builds a rule set [5,6]. Comprehensible classifiers are also learnt with the help of evolutionary algorithms (e.g., [7]). In recent years, a few Ant Colony Optimization (ACO) [8-10] based

algorithms have been proposed for the discovery of classification rules [11-15] which combine comprehensibility with higher predictive accuracy. ACO is one of the most famous meta-heuristic under the umbrella of swarm intelligence [16-18], which deals with the collective behavior of small and simple entities and has been used in many application domains. In the present paper, a recently proposed, state of the art ACO based classification rule discovery algorithm called CAntMiner, is improved and applied to twelve medical data sets widely tested in the machine learning community. The main objective here is to show that, for these problems, CAntMiner is able to achieve classification rates and generalization performance better than many other popular methods and algorithms.

The overall approach of the CAntMiner algorithm is sequential covering. It builds rules for all the classes present in the dataset. The algorithm starts with a full training set, creates a rule that covers a subset of the training data, adds this rule in the discovered rules list and removes the training samples that are correctly classified by the rule. This process continues until the training set is almost empty. The proposed algorithm has some unique features which allow it to achieve a higher accuracy rate as compared to some well known algorithms.

The remainder of this paper is organized as follows: in Section 2, we explain the basic concepts of ACO and give an overview of existing ACO algorithms for classification rules discovery; Section 3 describes our proposed approach and its improvements; in Section 4, we give the comparative results of ten classification algorithms on twelve publicly available data sets; finally, Section 5 concludes the paper and gives future directions of research.

2. **Ant Colony Optimization Metaheuristic and Classification Rules.**

Ant colony optimization is a branch of swarm intelligence (and, in general, a genre of population based heuristic algorithms [19]) inspired by the food behavior of biological ants. Ants inform other ants about the trail they are following by spreading a chemical substance called pheromone in the environment. Other ants that arrive in the vicinity are more likely to take the path with higher concentration of pheromone than the paths with lower concentrations.

This indirect form of information passing via the environment helps the ants to find the shortest path to the food source. An ant taking the shorter of the two paths will return earlier than the ant taking the longer path and hence the pheromone on the shorter path will be enhanced and subsequent ants will have a higher probability of taking the shorter path. Soon a new shorter path which bypasses the blockage will be established. The more ants follow a given trail, the more attractive that trail becomes and is followed by other ants.

This phenomenon has been modeled in the ACO meta-heuristic. An artificial ant constructs a solution to the problem by adding solution components one by one. After a solution has been constructed its quality (i.e., fitness) is determined and the components of the solution are assigned pheromone concentrations proportional to the quality. Subsequently other ants construct their solutions one by one and they are guided by the pheromone concentrations in their search for components to be added in their solutions. The components with higher pheromone concentrations are thus identified as contributing to a good solution and repeatedly appear in the solutions. It is expected that after a while the ants converge on a good, if not the optimal, solution.

For the application of ACO to a problem we need the following:

- The complete solution can be represented as a combination of different components.
- There should be a method to determine the fitness or quality of the solution.
- It is desirable, though not necessary, that there is a heuristic measure for the solution components.

Since its inception, ACO has been applied to solve many problems [8,17]. It is naturally suited to discrete optimization problems, such as quadratic assignment, job scheduling, subset problems, network routing, vehicle routing, load dispatch in power systems, bioinformatics and data mining [5,6,20] which is the subject of this paper.

2.1. **ACO for discovery of classification rules.** The ACO has been applied for discovery of classification rules. The first ACO based algorithm for classification rule discovery, called, AntMiner, was proposed by Parpinelli et al. [11]. An ant constructs a rule. It starts with an empty rule and incrementally constructs a rule by adding one term at a time. The selection of a term to be added is probabilistic and based on two factors: a heuristic quality of the term and the amount of pheromone deposited on it by the previous ants. The authors use the information gain as the heuristic value of a term. After the antecedent part of a rule has been constructed, the consequent of the rule is assigned by a majority vote of the training samples covered by the rule. The constructed rule is then pruned to remove the irrelevant terms and to improve its accuracy. The quality of the constructed rule is determined and pheromone values are updated on the trail followed by the ant proportional to the quality of rule. When all ants have constructed their rules then the best rule is selected and added to a discovered rule list. The training samples correctly classified by that rule are removed from the training set. This process is continued until the numbers of uncovered samples are less than the user specified threshold. The final product is an ordered discovered rule list that is used to classify the test data. AntMiner has been previously applied to medical datasets [21].

The extensions of the AntMiner algorithm are proposed by Liu et al. in AntMiner2 [12] and AntMiner3 [13,14]. In AntMiner2, the authors propose density estimation as a heuristic function instead of information gain used by AntMiner. They show that this simpler heuristic value does the same job as well as the complex one, and hence, AntMiner2 is computationally less expansive then the original AntMiner and has comparable performance. In AntMiner3, the authors use a different pheromone update method. They update and evaporate the pheromone of only those terms that occur in the rule and do not evaporate the pheromones of unused terms. In this way, exploration is encouraged.

Martens et al. [15] propose a Max-Min Ant System based algorithm (AntMiner+) that differs from the previously proposed AntMiners in several aspects. Only the iteration-best ant is allowed to update the pheromone, the range of the pheromone trail is limited within an interval, class label of a rule is chosen prior to the construction of the rule and a different rule quality measure is used. Other works on AntMiner include [22] in which an algorithm for discovering unordered rule sets has been presented. In [23], PSO algorithm is used for continuous valued attributes and ACO for nominal valued ones and these two algorithms are jointly used to construct rules. The issue of continuous attributes has also been dealt in [24,25]. An AntMiner version for multi-label classification problem can be found in [26]. Recently, an AntMiner based on Ant Colony System has been proposed [27].

Our proposed algorithm [28] is different in many ways from the previous AntMiner algorithms. The main novelty of our approach is a new class dependant heuristic function based on the compatibility of pair of terms which reduces the search space considerably and yields better results. This new heuristic function considers compatibility between recently added terms to better guide the ants to search through the environment. Accuracy of a classifier on medical data sets is a major concern, because this kind of data is very sensitive. The proposed CAntMiner yields better accuracy rate on most of the data sets.

3. **Compatibility Based AntMiner and Its Improved Version.** In this section, we first describe our recently proposed ACO based classification rules mining algorithm called CAntMiner. We begin with a general description of the algorithm and then discuss the heuristic function, pheromone update, rule pruning, early stopping, etc. We then propose some improvements in the algorithm.

3.1. **General description.** The core of the algorithm is the incremental construction of a classification rule of the type

IF <term1 AND term2 AND . . . >THEN <class>

by an ant. Each term is an attribute-value pair related by an operator. In our current experiments, we use "=" as the only possible relational operator. An example term is "Color = red". The attribute's name is "color" and "red" is one of its possible values. Since we use only "=" any continuous (real-valued) attributes present in the data have to be discretized in a preprocessing step.

For the ants to search for a solution, we need to define the space in which the search is to be conducted. The dimensions (or coordinates) of the search space are the attributes of the dataset. The possible values of an attribute constitute the range of values for the corresponding dimension in the search space. The task of the ant is to move in the search space in a forward direction and construct a rule, given a class label. It visits an attribute and chooses one of its possible values to form an antecedent condition of the rule (e.g., Color = red). The total number of terms for a dataset is equal to the

$$Total\_terms = \sum_{i=1}^{a} b_i \tag{1}$$

where a is the total number of attributes (excluding the class attribute) and b_i is the number of possible values that can be taken on by an attribute A_i. When an attribute has been visited, it cannot be visited again by an ant, because we do not allow conditions of the type "Color = Red OR Color = Green". The search space is such that an ant may pick the next term from any attribute and there is no ordering in which the attributes can be visited. An ant has to stop when all the attributes have been visited. It can also stop prematurely if the addition of the latest term makes the partial rule to cover training samples which only have the chosen class label. An example search space represented as a graph is shown in Figure 1.

In Figure 1, there are four attributes A, B, C and D having 3, 2, 3 and 2 possible values, respectively. An ant starts from the "Start" vertex and constructs a rule by adding conditions (attribute-value pairs or terms) for the antecedent part. After a term has been selected, all the other terms from the same attribute are prohibited for the ant. In this graph, the constructed path is C1, A3, D1 and B1.

A general description of the CAntMiner algorithm is shown in Figure 2. Its basic structure is that of Ant-Miner algorithm [11]. It is a sequential covering algorithm. It discovers a rule and the training samples correctly covered by this rule (i.e., samples which satisfy the rule antecedent and have the class predicted by the rule consequent) are removed from the training set. The algorithm discovers another rule using the reduced training set and after its discovery the training set is further reduced by removing the training samples covered by the newly discovered rule. This process continues until the training set is almost empty. The majority class label of the residual uncovered samples is used in a default rule added at the end of the discovered rule set.

One rule is discovered after one execution of the outer WHILE loop. First of all, a class label is chosen from the set of class labels present in the uncovered samples of training set. Each iteration of the REPEAT-UNTIL loop sends an ant to construct a rule. Each

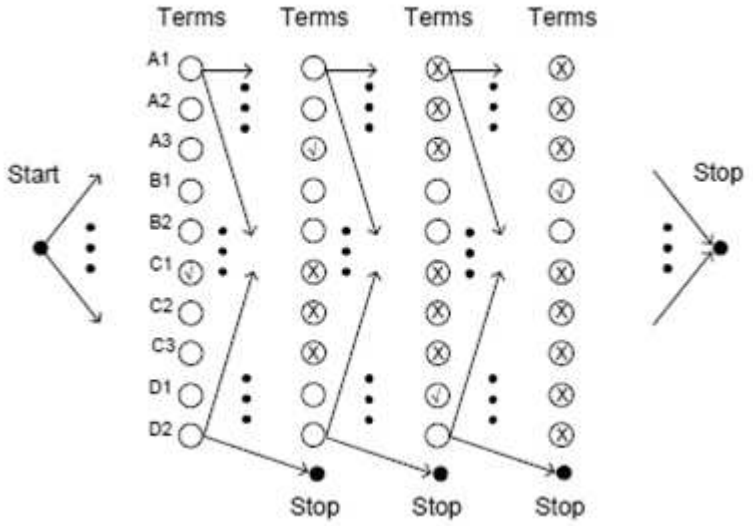

FIGURE 1. An example problem's search space represented as a graph

ant starts with an empty list of conditions and constructs the antecedent part of the rule by adding one term at a time. Every ant constructs its rule for the particular class chosen before beginning of the REPEAT-UNTIL loop. The quality of the rule constructed by an ant is determined and pheromones are updated and then another rule is constructed. Several rules are constructed through an execution of the REPEAT-UNTIL loop. The best one among them, after pruning, is added to the list of discovered rules. The algorithm terminates when the outer WHILE loop exit criterion is met. The output of the algorithm is an ordered set of rules. This set can then be used to classify unseen data. The main features of the algorithm are explained in detail in the following sub-sections.

3.2. **Rule construction.** The process of rule construction for a batch of ants starts by assigning them a class label. The ants construct their rules one by one, and each new ant is guided by the experience of the previous ants, available in the form of pheromone values. An ant constructs a rule by adding one term at a time. The choice of adding a term in the current partial rule is based on the pheromone values and heuristic values associated with the links from the previously added term to the candidate terms. An ant terminates its rule construction when all the instances covered by its rule are of the assigned class label or when there are no more attributes left for addition in the rule. The different aspects of rule construction are described below.

Class Commitment: A class is chosen probabilistically, by roulette wheel selection, on the basis of the weights of other classes present in the yet uncovered data samples for each iteration of the WHILE loop. The weight of a class is the ratio of its uncovered data samples to the total number of uncovered data samples. The class is chosen once and becomes fixed for the all the ant runs made in the REPEAT-UNTIL loop.

Pheromone Initialization: At the beginning of an iteration of the WHILE loop, the pheromone values on edges between all terms are initialized with the same amount of pheromone. The initial pheromone is

$$\tau_{ij}(t=0) = \frac{1}{\sum_{i=1}^{a} b_i} \tag{2}$$

```
Size (TrainingSet) = all training samples;
DiscoveredRuleList = ; // rule list is initialized with an empty list
while (TrainingSet >Max_uncovered_cases) do
  t = 1; // counter for ants
  j = 1; // counter for rule convergence test
  Select class label from the set of class labels;
  Set up the search space and initialize all links with the same amount of pheromone;
  Calculate heuristic values for all links;
  repeat
    Send an Antt which constructs a classification rule Rt for the selected class;
    Assess the quality of the rule and update the pheromone of all trails;
    if (R_t is equal to R_{t-1}) /*update convergence test*/ then
      j = j + 1;
    else
      j = 1;
    end if
    t = t + 1;
  until (t ≥ No_of_ants) OR (j ≥ No_rules_converg)
  Choose the best rule Rbest among all rules R_t constructed by all the ants;
  Prune the best rule Rbest;
  Add the pruned best rule Rbest to DiscoveredRuleList;
  Remove the training samples correctly classified by the pruned best rule Rbest;
end while
Add a default rule in the DiscoveredRuleList
```

FIGURE 2. The CAntMiner algorithm

where a is the total number of attributes (excluding the class attribute) and b_i is the number of possible values that can be taken on by an attribute A_i.

Term Selection: An ant incrementally adds terms in the antecedent part of the rule that it is constructing. The selection of the next term is subject to the restriction that the attribute A_i of that term should not be already present in the current partial rule. In other words, once a term has been included in the rule then no other term containing that attribute can be considered. Subject to this restriction, the probability of selection of a term for addition in the current partial rule is given by the equation:

$$p_{ij}(t) = \frac{\tau_{ij}(t).\eta_{ij}(s)}{\sum_{i=1}^{a} x_i \sum_{j=1}^{b_i} \{\tau_{ij}(t).\eta_{ij}(s)\}} \tag{3}$$

where $\tau_{ij}(t)$ is the amount of pheromone associated with the edge between $term_i$ and $term_j$ for the current ant (the pheromone value may change after the passage of each ant), $\eta_{ij}(s)$ is the value of the heuristic function associated with the edge for the current iteration s of the WHILE loop, a is the total number of attributes, x_i is a binary variable that is set to 1 if the attribute A_i has not been used by the current ant and else set to 0, and b_i is the number of values in the A_i attribute's domain. The denominator is used to normalize the numerator value for all the possible choices.

Heuristic Function: The heuristic value of a term gives an indication of its usefulness and thus provides a basis to guide the search. In order to guide the selection of next term, we use a heuristic function based on the compatibility of the most recently chosen term

with other candidate terms. The heuristic function is:

$$\eta_{k,i,j} = \frac{|term_j, class_k|}{|term_j|} * \frac{|term_i, term_j, class_k|}{|term_j, class_k|} \tag{4}$$

The most recently chosen term is $term_i$ and the term being considered for selection is $term_j$. The number of uncovered training samples having $term_i$ and $term_j$ and which belong to the committed class label k of the rule is given by $|term_i, term_j, class_k|$. This number is divided by the number of uncovered training samples which have $term_i$ and $classk$. The number of uncovered training samples having $term_i$ and which belong to the committed class label k of the rule is given by $|term_j, class_k|$. This number is divided by the number of uncovered training samples which have $term_j$.

Heuristic Function for the 1st term: The heuristic value when considering the 1st term of the rule antecedent is calculated on the basis of the following equation:

$$\eta_{k,start,j} = \frac{|term_j, class_k| + 1}{|term_j| + Total\_classes} \tag{5}$$

Rule Termination: An ant continues to add terms in the rule it is constructing and stops only when all the samples covered by the rule have homogenous class label (which is committed prior to the REPEAT-UNTIL loop). The other possibility of termination is that there are no more terms to add. Due to these termination criteria it might happen that a constructed rule may cover only one training sample.

3.3. **Rule quality and pheromone update.** When an ant has completed the construction of a rule its quality is calculated. The quality, Q, of a rule is computed by using confidence and coverage of the rule and is given by:

$$Q = \frac{TP}{Covered} + \frac{TP}{N} \tag{6}$$

where TP is the number of samples covered by the rule that have the same class label as that of the rule's consequent, Covered is the total number of samples covered by the rule, and N is the number of samples in the training set yet uncovered by any rule in the discovered rule set. The second portion is added to encourage the construction of rules with wider coverage.

The pheromone values are updated so that the next ant can make use of this information in its search. The amount of pheromone on the links between consecutive terms occurring in the rule is updated according to the equations:

$$\tau_{ij}(t) = (1 - \rho)\tau_{ij}(t) \tag{7}$$

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \left(1 - \frac{1}{1+Q}\right)\tau_{ij}(t) \tag{8}$$

where $\tau_{ij}(t)$ is the pheromone value encountered by Ant_t between $term_i$ and $term_j$. The pheromone evaporation rate is represented by ρ and Q is the quality of the rule constructed by Ant_t.

3.4. **Termination of REPEAT-UNTIL loop.** The REPEAT-UNTIL loop is used to construct as many rules as the user defined number of ants. After the construction of each rule its quality is determined and the pheromones on the trails are updated accordingly. The pheromone values guide the construction of next rule. An early termination of this loop is possible if the last few ants have constructed the same rule. This implies that the pheromone values on a trail have become very high and convergence has been achieved. Any further rule construction will most probably yield the same rule again. Hence, the loop is terminated prematurely. For this purpose, each constructed rule is compared with

the last rule and a counter is incremented if both the rules are the same. If the value of this counter exceeds a threshold "No. of rules converged", then the loop is terminated. In our experiments, we use a value of 10 for this threshold.

3.5. **Pruning of best rule.** Rule pruning is the process of finding and removing any irrelevant terms that might have been included in the constructed rule. The rule pruning procedure starts with the full rule. It temporarily removes the first term and determines the quality of the resulting rule. It then replaces the term back and temporarily removes the second term and again calculates the quality of the resulting rule. This process continues until all the terms present in the rule are dealt with. After this assessment, if there is no term whose removal improves or maintains the quality then the original rule is retained. However, if there is one or more terms whose removal improves (or maintains) the quality of the rule then the term whose removal most improves the quality of the rule is permanently removed. In such cases, the shortened rule is again subjected to the procedure of rule pruning. The process continues until any further shortening is impossible (removal of any term present in the rule leads to decrease in its quality) or if there is only one remaining term in the rule.

3.6. **Final rule set.** The best rule is placed in the discovered rule set after pruning and the training samples correctly covered by the rule are removed and have no role in the discovery of other rules. A new iteration starts for discovery of the next rule if the stoppage criterion of the algorithm is not met. A final default rule is added at the bottom of the rule set. The class label for this rule is the majority class label in the set of uncovered training cases at the stoppage of the algorithm.

Stoppage of Algorithm: We continue the algorithm until the training dataset is almost empty. For this purpose, a threshold, called "Max_uncovered_samples" is specified. If, after an iteration of the REPEAT-UNTIL loop, the remaining samples in the training set are equal to or below this specified threshold, then stop the algorithm.

Use of Discovered Rule Set for Classifying Unseen Samples: A new test sample unseen during training is classified by applying the rules in order of their discovery. The first rule whose antecedents match the new sample is fired and the class predicted by the rule's consequent is assigned to the sample. If none of the discovered rules are fired then the final default rule gets activated.

3.7. **Improvements in CAntMiner.** We have changed the CAntMiner algorithm in two important ways:

- **Stopping Criterion of the Algorithm**: The algorithm keeps on discovering rules until the last remaining samples are of homogenous class labels. Previously, the algorithm stopped when the remaining samples fell below or became equal to a user specified number (called Max_uncovered_cases). The default rule added at the end of the discovered rule set has the class label of last remaining samples (which are of homogenous class labels).
- **Relative Importance of Pheromone and Heuristic**: Parameters alpha and beta, are used to control the relative importance of pheromone and heuristic in the probabilistic determination of next term chosen by an ant in the process of rule construction (Equation (3)). Previously, we did not use alpha and beta which was equal to having their values set to one.

Stopping Criterion of the Algorithm: In CAntMiner, the execution of the algorithm is continued until the training dataset is almost empty. To define 'almost empty' a threshold has to be defied on the number of uncovered training samples present in the data set. If, after an iteration of the REPEAT-UNTIL loop, the remaining uncovered samples

are equal to or below this specified threshold, then the algorithm is stopped. However, alternate approaches can be also be implemented to accommodate early stopping of the algorithm. In this version, the algorithm continue till the training set is completely empty or has remaining uncovered training samples of only one class label. This does not require a user defined parameter and also does not cause the information present in the remaining uncovered training samples to be thrown away. The disadvantage of our approach is that it may lead to rules which cover one or two training samples and may cause over-fitting to the data.

In the improved CAntMiner, we keep on discovering new rules until all the remaining training samples have the same class label, and then add a default rule with that class label. The results of comparison between these two alternate methods on a suite of 12 datasets (other than medical datasets used in Section 4 below) show that the accuracy improves in 8 out of 12 datasets for the improved algorithm.

Relative Importance of Pheromone and Heuristic: By experiments, we have found that the relative importance of pheromone and heuristic in Equation (3) is not the same. We now use the following equation:

$$p_{ij}(t) = \frac{{\tau_{ij}}^{\alpha}(t).{\eta_{ij}}^{\beta}(s)}{\sum_{i=1}^{a} x_i \sum_{j=1}^{b_i} \{{\tau_{ij}}^{\alpha}(t).{\eta_{ij}}^{\beta}(s)\}} \tag{9}$$

We have experimented with different values of alpha and beta in Equation (9). The value of alpha is varied from 0 to 3 and for each of these values the value of beta is set at 1. Then the value of alpha is set at 1 and beta is varied from 1 to 4. This gives us the following ratios for alpha/beta: 3, 2, 1, 0, 0.5, 0.33 and 0.25. The best predictive accuracy results are obtained when alpha = 1 and beta = 3. This proved that the incorporation of alpha and beta is a useful addition in the algorithm. The suite of 12 datasets for this experiment was other than that of the medical datasets used in Section 4 below.

4. **Experiments and Analysis.**

In this section, we report our experiments and the results obtained.

4.1. **Datasets and performance metrics and parameter settings.**

In our experiments, we use a suite of twelve medical datasets obtained from the UCI repository [29] to validate the usefulness of the CAntMiner algorithm. A brief description of the datasets is given below.

Data Set	Attributes	Instances	Classes
Blood Transfusion	4	748	2
Breast Cancer Wisconsin (Diagnostic)	31	569	2
Breast Cancer Wisconsin (Original)	10	699	2
Dermatology	34	366	6
Ecoli	8	336	8
Haberman's Survival	3	306	2
Heart disease-Cleveland	13	303	5
Hepatitis	19	155	2
Lung Cancer	56	32	3
Mammographic Mass	5	961	2
Pima Indian Diabetes	8	768	2
SPECT (Heart)	22	267	2

CAntMiner algorithm works with categorical attributes and continuous attributes need to be discretized in a preprocessing step. We use unsupervised discretization filter of Weka-3.4 machine learning tool [30] for discretizing continuous attributes in all our experiments. This filter first computes the intervals of continuous attributes from the training dataset and then uses these intervals to discretize them.

Our performance metric for the discovered rule set is its predictive accuracy. The predictive accuracy is defined as the percentage of testing samples correctly classified by the classifier. 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. CAntMiner has been implemented by us in Matlab.

4.2. **Parameter settings.** CAntMiner has following user defined parameters.

- number of ants,
- number of rules converged (for early termination of REPEAT-UNTIL loop),
- the powers of pheromone and heuristic values (α, β) in the probabilistic selection Equation (9),
- the pheromone evaporation rate (used in Equation (7)).

If the value of the 'Number of Ants' parameter is set at 1000, then a maximum of 1000 rules can be constructed out of which the best one is chosen to be placed in the discovered rule set. It has been observed that very less number of ants are used, on the average, because the REPEAT-UNTIL loop gets terminated if convergence is achieved (all of the recently discovered rules are duplicates of each other). For this purpose, the threshold parameter 'Number of rules converged' (10 in our experiments) is used. The number of ants is not a sensitive parameter due to the early convergence option of the REPEAT-UNTIL loop and any high number (e.g., 1000 used by us) would serve adequately. We have observed that the average of the actual number of ants used per execution of REPEAT-UNTIL loop is much less than the maximum number allowed (i.e., 1000). Convergence speed is an important aspect, particularly, for large and high dimensional datasets.

The relationship between α, β and ρ is complex and needs to be analyzed experimentally. In order to get some indication of the influence of α, β and ρ on the performance of CAntMiner, different combinations of these parameters were used and the accuracy results were analyzed. The suite of datasets used was other than medical datasets listed above. We found out that accuracy is highest when $\rho = 0.15$, $\alpha = 1$ and $\beta = 3$.

4.3. **Accuracy of results.** We compare the results of our algorithm with those for AntMiner, AntMiner2, AntMiner3, C4.5 [31,32], Ripper, AdaBoost, k-nearest neighbor, logistic regression, naive Bayes and support vector machines (SVM) [33]. AntMiner has been implemented by us in Matlab. For other algorithms, we use the Weka machine learning tool [30].

As stated before in Section 4.1, we use unsupervised discretization filter of Weka-3.4 machine learning tool [30] for discretizing continuous attributes as a preprocessing step.

The CAntMiner has five user defined parameters: number of ants, evaporation rate, convergence counter, alpha and beta. The values of these parameters are given in Table 1. The same parameters have been retained while obtaining results for previous AntMiners. Some of these values have been determined experimentally and others have been chosen because they seem reasonable and have been used by other AntMiner versions reported in literature [11-15].

The predictive accuracies of the compared algorithms with standard deviations are shown in Tables 2 and 3. Ten fold cross validation is used to obtain the results.

TABLE 1. Parameters used in experiments

Parameter	Value
Number of Ants	1000
Max. uncovered cases (used in other Ant Miners)	10
Min. cases per rule (used in other Ant Miners)	10
Evaporation rate	0.15
No. of rules converged	10
Alpha	1
Beta	3

TABLE 2. Comparison with algorithms which discover rule sets, average predictive accuracies are obtained using 10-fold cross validation

Data Sets	CAnt Miner	Ant-Miner	Ant Miner 2	Ant Miner 3	C4.5	Ripper
Blood Transfusion	**79.57 ± 3.04**	77.30 ± 6.37	79.44 ± 3.64	77.56± 3.31	77.71 ± 12.32	76.11 ± 14.54
Breast Cancer – W (Diag)	87.33 ± 5.46	85.26± 4.22	86.66 ± 4.56	88.05 ± 5.40	93.31 ± 2.72	**94.03 ± 3.72**
Breast Cancer – W (Org)	**97.85 ± 1.69**	94.64 ± 2.74	92.70 ± 2.82	93.56 ± 3.45	94.84 ± 2.62	95.57 ± 2.17
Dermatology	94.27 ± 4.51	58.72 ± 7.36	59.18 ± 14.91	67.47 ± 9.57	**95.07 ± 2.80**	93.96 ± 3.63
Ecolli	**83.64 ± 6.11**	47.52 ± 11.32	43.09 ± 9.76	44.61 ± 10.32	82.99 ± 7.72	79.18 ± 2.22
Haberman's survival	**83.05 ± 7.80**	71.99 ± 7.57	73.94 ± 5.33	74.72 ± 4.62	73.88 ± 4.66	72.47 ± 6.74
Heart Disease – Cl	80.74 ± 9.37	80.74 ± 4.94	78.15 ± 10.25	**87.78 ± 6.77**	78.43 ± 6.26	73.59 ± 9.57
Hepatitis	**87.17 ± 7.81**	80.67 ± 8.67	81.46 ± 11.89	80.17 ± 10.23	68.25 ± 11.63	73.46 ± 8.21
Lung cancer	87.50 ± 16.32	63.33 ± 6.48	84.17 ± 14.54	79.68 ± 15.32	**97.50 ± 7.50**	94.17 ± 12.45
Mammographic – Mass	78.67 ± 4.65	78.25 ± 3.48	79.40 ± 3.64	82.33 ± 3.56	82.44 ± 4.56	**83.52 ± 4.52**
Pima Indian Diabetes	**80.26 ± 6.19**	74.63 ± 6.65	72.56 ± 4.46	70.88 ± 5.05	72.11 ± 6.96	77.96 ± 7.47
SPECT (Heart)	**87.41 ± 4.97**	75.38 ± 5.50	83.56 ± 8.34	78.68 ± 5.17	80.03 ± 51.85	79.29 ± 19.43

TABLE 3. Comparison with algorithms which do not discover rule sets, average predictive accuracies are obtained using 10-fold cross validation

Data Sets	CAnt Miner	AdaBoost	kNN	Logit	NaiveBayes	SVM
Blood Transfusion	**79.57 ± 3.04**	75.84 ± 16.0	71.01 ± 14.98	77.17 ± 14.49	72.37 ± 17.22	76.24 ± 15.33
Breast Cancer – W (Diag)	87.33 ± 5.46	94.71 ± 2.89	95.06 ± 2.88	95.24 ± 2.07	93.48 ± 4.49	**97.72 ± 1.45**
Breast Cancer – W (Org)	**97.85 ± 1.69**	94.83 ± 1.67	96.42 ± 1.54	96.56 ± 1.21	96.13 ± 1.19	96.70 ± 0.69
Dermatology	94.27 ± 4.51	96.45 ± 2.88	95.62 ± 3.22	96.71 ± 2.83	96.98 ± 2.75	**97.54 ± 2.99**
Ecolli	83.64 ± 6.11	81.53 ± 7.25	80.86 ± 6.45	**86.31 ± 5.38**	85.66 ± 2.83	82.69 ± 5.04
Haberman's survival	**83.05 ± 7.80**	69.76 ± 7.21	70.11 ± 11.95	73.08 ± 3.99	74.05 ± 4.76	73.44 ± 0.97
Heart Disease – Cl	80.74 ± 9.37	82.33 ± 6.08	80.71 ± 6.17	77.0 ± 5.05	**85.19 ± 8.27**	80.32 ± 6.25
Hepatitis	**87.17 ± 7.81**	74.71 ± 15.02	67.62 ± 9.06	64.25 ± 8.87	74.79 ± 6.90	75.37 ± 8.62
Lung cancer	87.50 ± 16.32	**94.17 ± 12.45**	84.17 ± 16.87	90.83 ± 16.93	87.50 ± 16.31	**94.17 ± 12.45**
Mammographic – Mass	78.67 ± 4.65	81.88 ± 4.54	78.07 ± 5.99	**82.92 ± 3.91**	82.69 ± 3.82	80.32 ± 4.37
Pima Indian Diabetes	**80.26 ± 6.19**	74.33 ± 6.49	69.75 ± 7.13	77.98 ± 5.91	75.49 ± 4.50	77.06 ± 4.09
SPECT (Heart)	**87.41 ± 4.97**	84.52 ± 15.29	81.20 ± 16.77	81.94 ± 15.92	76.87 ± 12.72	87.20 ± 11.48

The results indicate that the CAntMiner achieves higher accuracy rate than the compared algorithms for most of the datasets. The better results are due to new heuristic function and stopping criteria of a rule construction process. For making comparison fair, we used same parameters for all versions of Ant Miners. The compared algorithms contain both comprehensible and non-comprehensible (statistical) classifiers.

5. **Conclusions and Future Work.** Almost a decade after the first attempt to apply an ACO based classification rule discovery algorithm to medical data sets [21], this paper proposes improvements in a recently proposed algorithm whose main highlight is the use of a novel heuristic function and reports its application to medical datasets. The experimental results show that the proposed approach achieves higher accuracy rate than other conventional algorithms.

CAntMiner outputs a set of rules which are comprehensible and can be made part of a larger decision support system [34]. However, there are some issues pertaining to comprehensibility, three of which are discussed in this section. These are some of the many possible directions in which the present work can be extended.

CAntMiner is designed primarily for categorical data. Real valued attributes are discretized in a preprocessing step. A better discretization algorithm is more likely to facilitate better rule discovery. CAntMiner can be modified such that the job of finding discretization intervals can be assigned to it. However, this would increase the time complexity of the algorithm. Another issue pertaining to comprehensibility is that CAntMiner discovers an ordered set of rules. This means that a rule has to be understood in conjunction with the rules preceding it. This constraint becomes cumbersome in cases of large rule sets. A solution to this problem is the discovery of unordered rule set. This can be achieved by running the algorithm for one of the classes until the training set is (almost) empty of samples of that class. The original training set is then restored and rules for another class are discovered. This process continues until rules for all classes have been discovered. Rules discovered in such a manner can be applied in an unordered manner and all can be analyzed independently from one another.

If expert knowledge exists about a problem domain then it is desirable to incorporate it in the classifier learning process. This has two potential advantages: the learning process may speed up and the learning is less likely to be contradictory to the known facts. In CAntMiner, domain knowledge can be incorporated in two ways. First, the links between any two conditions which cannot occur together can be deleted from the search space. This can be done by assigning an initial pheromone value of zero on that link. For example, if it is known that a disease cannot occur after 12 years of age then we can eliminate all the links to and from the term "Age $>$ 12 years" in the search space set up for constructing the rules with class label "Disease = Yes". Secondly, initial pheromone values can be increased and decreased to reflect the domain knowledge. If two terms are known to occur together we can increase the initial pheromone values on the corresponding links and vice versa. Domain knowledge can also be incorporated after the CAntMiner algorithm has delivered its rule set. Rules not confirming to the known knowledge can be modified or removed. Furthermore, rules can be generalized or made more specific and even new rules can be added. In this regards decision tables are helpful.

REFERENCES

[1] G. Shobha and S. Sharma, Knowledge discovery for large data sets using artificial neural network, *International Journal of Innovative Computing, Information and Control*, vol.1, no.4, pp.635-642, 2005.

[2] M. Brameier and W. Banzhaf, A comparison of linear genetic programming and neural networks in medical data mining, *IEEE Transactions on Evolutionary Computation*, vol.5, no.1, 2001.

[3] C. Lin, C. Chen and C. Lee, Classification and medical diagnosis using wavelet-based fuzzy neural networks, *International Journal of Innovative Computing, Information and Control*, vol.4, no.3, pp.735-748, 2008.

[4] D. Nauck and R. Kruse, Obtaining interpretable fuzzy classification rules from medical data, *Artificial Intelligence in Medicine*, vol.16, no.2, 1999.

[5] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd Edition, Morgan Kaufmann Publishers, 2006.
[6] R. O. Duda, P. E. Hart and D. G. Stork, *Pattern Classification*, Wiley, 2000.
[7] I. de Falco, A. D. Cioppa, A. Iazzetta and E. Tarantino, An evolutionary approach for automatically extracting intelligible classification rules, *Knowledge and Information Systems*, vol.7, no.2, 2005.
[8] M. Dorigo and T. Sttzle, *Ant Colony Optimization*, MIT Press, Cambridge, MA, 2004.
[9] M. Dorigo, V. Maniezzo and A. Colorni, Ant system: Optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernatics, Part B*, vol.26, no.1, 1996.
[10] M. Dorigo and L. M. Gambardella, Ant colony system: A cooperative learning approach to the travelling salesman problem, *IEEE Transactions on Evolutionary Computation*, vol.1, no.1, 1997.
[11] R. S. Parpinelli, H. S. Lopes and A. A. Freitas, Data mining with an ant colony optimization algorithm, *IEEE Transactions on Evolutionary Computation*, vol.6, no.4, pp.321-332, 2002.
[12] B. Liu, H. A. Abbass and B. McKay, Density-based heuristic for rule discovery with ant-miner, *Proc. of the 6th Australasia – Japan Joint Workshop on Int. Evol. Syst.*, Canberra, Australia, pp.180-184, 2002.
[13] B. Liu, H. A. Abbass and B. McKay, Classification rule discovery with ant colony optimization, *Proc. of IEEE/WIC Int. Conf. Int. Agent Technol.*, pp.83-88, 2003.
[14] B. Liu, H. A. Abbass and B. McKay, Classification rule discovery with ant colony optimization, *IEEE Computational Intelligence Bulletin*, vol.3, no.1, 2004.
[15] D. Martens, M. de Backer, R. Haesen, J. Vanthienen, M. Snoeck and B. Baesens, Classification with ant colony optimization, *IEEE Transactions on Evolutionary Computation*, vol.11, no.5, 2007.
[16] A. P. Engelbrecht, *Computational Intelligence, An Introduction*, 2nd Edition, John Wiley and Sons, 2007.
[17] A. P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, John Wiley and Sons, 2005.
[18] J. Kennedy, R. C. Eberhart and Y. Shi, *Swarm Intelligence*, Morgan Kaufmann/Academic Press, 2001.
[19] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing, Natural Computing Series*, 2nd Edition, 2007.
[20] A. Abraham, C. Grosan and V. Ramos, Swarm intelligence in data mining, *Studies in Computational Intelligence*, vol.34, 2006.
[21] R. Parpinelli, H. Lopes and A. Freitas, An ant colony based system for data mining: Applications to medical data, *Proc. of 2001 Genetic and Evolutionary Computation Conf.*, pp.791-798, 2001.
[22] J. Smaldon and A. A. Freitas, A new version of the Ant-Miner algorithm discovering unordered rule sets, *Proc. of Genetic and Evolutionary Computation Conf.*, pp.43-50, 2006.
[23] N. Holden and A. A. Freitas, A hybrid PSO/ACO algorithm for classification, *Proc. of GECCO-2007 Workshop on Particle Swarms: The 2nd Decade*, ACM Press, New York, 2007.
[24] S. Swaminathan, *Rule Induction Using Ant Colony Optimization for Mixed Variable Attributes*, Master Thesis, Texas Tech. University, 2006.
[25] F. Otero, A. Freitas and C. Johnson, cAnt-Miner: An ant colony classification algorithm to cope with continuous attributes, *Ant Colony Optimization and Swarm Intelligence, LNCS*, vol.5217, 2008.
[26] A. Chan and A. Freitas, A new ant colony algorithm for multi-label classification with applications in bioinformatics, *Proc. of Genetic and Evolutionary Computation Conference*, 2006.
[27] U. Boryczka and J. Kozak, New algorithms for generation decision trees – Ant-Miner and its modifications, *Foundations of Computational Intelligence*, vol.6, 2009.
[28] W. Shahzad and A. R. Baig, Compatibility as a heuristic for construction of rules by artificial ants, *Journal of Circuits, Systems, and Computers*, vol.19, no.1, 2010.
[29] S. Hettich and S. D. Bay, *The UCI KDD Archive. Irvine*, Department of Inf. Comput. Sci., University California, CA, http://kdd.ics.uci.edu, 1996.
[30] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd Edition, Morgan Kaufmann, 2005.
[31] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.
[32] J. R. Quinlan, Generating production rules from decision trees, *Proc. of Int. Joint Conf. Artificial Intelligence*, San Francisco, 1987.
[33] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, New York, 1995.
[34] Y. Min, D. Kim and T. Lim, A clinical practice guided decision support system based on medical ontology, *International Journal of Innovative Computing, Information and Control*, vol.6, no.5, pp.2361-2369, 2010.