Vandana Publications

Volume-4, Issue-1, February-2014, ISSN No.: 2250-0758
International Journal of Engineering and Management Research
Available at: www.ijemr.net
Page Number: 5-8

Classification Rule Mining Based on Ant Colony Optimization

Jyoti[1], Sakshi[2], Divesh[3]
[1,2,3]Department of Computer Science & Engineering, Guru Jamabeshwar University of Science and Technology, Hissar, INDIA

ABSTRACT

Classification rule mining is an important function of data mining, and is applied in many data analysis tasks. The classification rule mining algorithm based-on ant colony optimization (ACO) is researched in this paper. Some improvements are implemented based on existing research to enhance classification predictive accuracy and simplicity of rules. Ant-based algorithms or ant colony optimization (ACO) algorithms have been applied successfully to combinatorial optimization problems. The algorithm is realized under the GALIB247 tool.

Keywords: **Ant Colony Optimization, Data Mining, Knowledge Discovery, Classification.**

I. INTRODUCTION

Need for automatic discovery of hidden knowledge in large databases has led to the emergence of a new field called Knowledge Discovery in Databases (KDD). KDD is nontrivial process of identifying valid, novel, potentially useful and ultimately understandable patterns in data [1]. Data mining is particular step in KDD process to automate the process of extracting knowledge from databases[1]. Data mining explores large amount of data in order to find hidden patterns and/or relations between attributes, subsequently validates the findings by applying the detected pattern to a test data[1][2].

Out of several data mining tasks, including regression, clustering, dependence modeling, etc, classification is most studied and popular data mining task. In classification task each example (instance) in the dataset consists of two parts- a set of predictor attributes and a goal attribute namely the class of the instance. The main objective of classification is to build a model that predicts the class of an unseen data instance through predicting attributes. Classification rules are extracted in the form '*If-Then*' rules by automated learning of the classification model[3].Classification rules have the advantage of representing knowledge at a high level of abstraction, so that they are intuitively comprehensible to the users. Classification rules are commonly represented as follows:

IF (cond1) AND (cond2) AND (cond3…..) AND (cond n) THEN (predicted value)

The rule antecedent (IF part) contains a set of conditions, usually connected by a logical conjunction operator (AND). In proposed work, each rule condition is referred as a term, so that the rule antecedent is a logical conjunction of terms in the form: IF *term1* AND *term2* AND ... Each term is represented as a triple <*attribute*, *operator*, *value*>, for instance<Gender = female>. The rule consequent specifies the class predicted for the cases whose predictor attributes satisfy all the terms specified in the rule antecedent. From a data mining viewpoint, this kind of knowledge representation has the advantage of being intuitively comprehensible for the user, as long as the number of discovered rules and the number of terms in rule antecedents are not large[3].

Swarm intelligence deals with the collective behavior of small and simple entities and has been used in many optimization application domains. It is an intelligent, innovational and distributed paradigm for solving optimization problems. Recent research studies suggest that both data mining and swarm intelligence can be used together for wide range of real world data mining problems including classification, clustering, regression, and image processing [5][6]. Ant colony optimization (ACO) is a famous technique under the umbrella of swarm intelligence proposed by M. Dorigo *et al*in 1992 in his PhD thesis as a meta-heuristic method for solving a range of optimization problems. It is naturally suited to discrete optimization problems, such as, quadratic assignment, job scheduling, subset problems, network routing, vehicle routing, graph coloring problem, bio informatics and data mining. ACO algorithms involve simple agents (ants) that cooperate with one another to achieve an emergent, unified behavior for the system as a whole, producing a robust system capable of finding high-quality solutions for problems with a large search space. In the context of rule discovery, an ACO algorithm has the ability to perform a flexible, robust search for a good combination of terms (logical conditions) involving values of the predictor attributes[4]. This paper proposes an ACO (Ant Colony Optimization) approach to discover Classification Rules from datasets. The proposed work entails a slight modification in Ant Miner algorithm along with the application of algorithm on three datasets.

The remainder of the paper is organized as follows. Section 2 gives the background details for necessary contextualization of work that gives the basic idea of the ant colony system. In section 3, Ant Miner classification technique is described. Section 4 reports the experimental results evaluating the performance of algorithm. Conclusion and the future directions are presented in Section 5.

II. BACKGROUN DETAILS AND RELATED WORK

2.1 Ant System

Ant Colony Optimization (ACO) technique is inspired by the behavior of real ant colonies. When searching for food, ants initially explore the surrounding area, leaving chemical evidence on the path that it took in order to be followed by other ants. As an ant finds food, it evaluates the quality and the quantity of it and carries some of them back to the nest dropping pheromone in amounts proportional to the quality and the quantity of the founded food. The pheromone trial will probabilistically guide the ants to the food source. Eventually, this kind of behavior will lead the convergence of the ant taking the shortest path tothe best food source[4].The main steps of (ACO) algorithms are as follows:

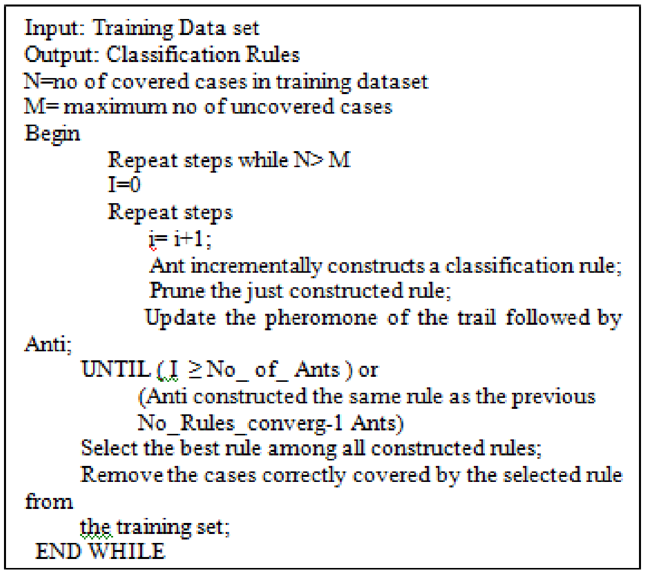
1. Pheromone trail initialization
2. Solution construction using pheromone trail: eachant constructs a complete solution to the problem according to a probabilistic model
3. Solution evaluation: evaluate the quality of the solution based on a problem specific fitness function.
4. Pheromone trail update: this is applied in two phases, where each ant deposits an amount of pheromone which is proportional to the fitness of its solution, and evaporation, where a fraction of the pheromone evaporates in order to avoid stagnation.

In essence, the design of an ant system implies the specification of the following aspects:

• An environment that represents the problem domain in such away that it is suitable for the ants to navigate and construct a solution for the problem.
• A problem dependent heuristic evaluation function (η), which represents a quality factor for the different steps that construct the solution.
• A rule for pheromone updating (τ), which takes into account the evaporation and there enforcement of the trails.
• A probabilistic transition rule based on the value of the heuristic function (η) and on the strength of the pheromone trail (τ) that is used to iteratively construct a solution.
• A fitness function by which the constructed solution is evaluated.
• A clear specification of when the algorithm converges to a solution[8][10].

2.2 Ant Miner Classification Algorithm

In an ACO algorithm each ant incrementally constructs/modifies a solution for the target problem. Ant-Miner is an ACO based classification algorithm that follows a sequential covering approach to discover classification rules. The discovered classification rules cover all, or almost all, the training cases. The algorithm shown in Fig.1 describes the working of the Ant Miner technique to find out the best rules from given data set. [ref of freit as paper]



```
Input: Training Data set
Output: Classification Rules
N=no of covered cases in training dataset
M= maximum no of uncovered cases
Begin
        Repeat steps while N> M
        I=0
        Repeat steps
            i= i+1;
            Ant incrementally constructs a classification rule;
            Prune the just constructed rule;
            Update the pheromone of the trail followed by
Anti;
    UNTIL (I ≥ No_ of_ Ants ) or
            (Anti constructed the same rule as the previous
            No_Rules_converg-1 Ants)
    Select the best rule among all constructed rules;
    Remove the cases correctly covered by the selected rule
from
    the training set;
 END WHILE
```

Fig.1 Ant Miner algorithm

Following steps are applied for rule discovery using Ant Miner

A.Basic operation

In core operation of Ant-Miner the REPEAT-UNTIL loop of Algorithm, the current ant iteratively adds one term at a time to its current partial rule. Let *termij* be a rule condition of the form *Ai* = *Vij*, where *Ai* is the *i*-th attribute and *Vij* is the *j*-th value of the domain of *Ai*. The probability that *termij* is chosen to be added to the current partial rule is given by Equation:

$$p_{ij} = \frac{\eta_{ij} \cdot \tau_{ij}(t)}{\sum_{i=1}^{a} x_i \cdot \sum_{j=1}^{b_i} \left(\eta_{ij} \cdot \tau_{ij}(t)\right)}$$

where:

- *Ηij* is the value of a problem-dependent heuristic function for *termij*. The higher the value of Ƞ*ij* indicates the higher its probability of being chosen.
- $\tau_{ij}(t)$ tis the amount of pheromone for each *termij* at iteration *t*, corresponding to the amount of pheromone currently available in the position *i,j* of the path being followed by the current ant.
- *a* is the total number of attributes.
- *xi* =1 if the attribute *Ai* was not yet used by the current ant, 0 otherwise.
- *bi* is the number of values in the domain of the *i*-th attribute.

B. Heuristic Function

For each *termij* that can be added to the current rule, Ant-Miner computes the value h ij of a heuristic function that is an estimate of the quality of this term, with respect to its ability to improve the predictive accuracy of the rule. The quality here is measured in terms of the entropy for preferring this term to the others, and is given by the Equation:

$$\eta_{ij} = \frac{\log_2 k - \left(W \middle| A_i = V_{ij}\right)}{\sum_{i=1}^{a} x_i \cdot \sum_{j=1}^{b_i} \left(\log_2 k - H\left(W \middle| A_i = V_{ij}\right)\right)}$$

where W is the class attribute (i.e., the attribute whose domain consists of the classes to be predicted). k is the number of classes and $H(W|A_i = V_{ij})$ is measure of entropy associated with that term.

C.Rule Pruning

Rule pruning potentially increases the predictive power of the rule, helping to avoid its over fitting to the training data. Another motivation for rule pruning is that it improves the simplicity of the rule, since a shorter rule is usually easier to be understood by the user than a longer one. The basic idea is to iteratively remove one-term-at-a-time from the rule while this process improves the quality of the rule [7][9].

D.Pheromone Updating

At each iteration of the WHILE loop of algorithm, all *termij* are initialized with the same amount of pheromone, so that when the first ant starts its search, all paths have the same amount of pheromone. The initial amount of pheromone deposited at each path position is inversely proportional to the number of values of all attributes, and is defined by Equation:

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

where *a* is the total number of attributes, and *bi* is the number of possible values that can be taken on by attribute *Ai*[7][9].

III. PROPOSED APPROACH

This section proposes a little modification in basic Ant miner algorithm. The proposed modification is carried out in rule pruning step of the algorithm. A rule with fix number of terms is discovered rather than to discover a long rule and then prune it. This is more efficient in terms of number of database scans. A long rule approach requires K^3*N number of database scans whereas proposed approach requires only K*N number of database scans. Here K is the number of attributes and N is the size of database. Rests of the steps of proposed algorithm are same as in basic Ant Miner Algorithm. Later this algorithm is also applied on different datasets for comparison with other rule discovery algorithms.

Rule construction is commenced form navigation of antfrom node to node. Ant transition from node to another is probabilistic based on the pheromone amount on the edge between the two nodes and the value calculated by a heuristic function applied on the destination node. After a rule is constructed, it is evaluated against the training set, and the pheromone on the rule edges is updated based on the quality of the rule. In each iteration, the best rules generated are selected and added to the result rule set, the covered cases in the training set by the rules are removed and the pheromone in the construction graph is reset. The algorithm runs until percentage of the cases are covered or for a specific number of iterations.

3.1 Use of the Discovered Rules for Classifying New Cases

In order to classify a new test case, unseen during training, the discovered rules are applied in the order they are discovered (recall that discovered rules are kept in an ordered list). The first rule that covers the new case is applied – that is, the case is assigned the class predicted by that rule's consequent. It is possible that no rule of the list covers the new case. In this situation the new case is classified by a default rule that simply predicts the majority class in the set of uncovered training cases, this is, the set of cases that are not covered by any discovered rule.

IV. EXPERIMENTAL RESULTS

The performance of modified Ant-Miner is evaluated using various public-domain data sets from the UCI (University of California at Irvine) repository [13]. The experimentation is implemented using GALIB247, on Ubuntu platform. The main characteristics of the data sets used in experiment are summarized in Table 1.

Table 1.DataSet Used in Experiment

Data Set	Cases	Attributes	Classes
Tic_Tac_Toe	958	9	3
Vote	232	16	2
Mushroom	5610	22	2

Table 2 is describing the performance of Ant-Miner by comparing it with J48(decision tree) and ripper classification, well-known classification-rule discovery algorithms in terms of accuracy rates and simplicity of the rule sets produced on the four datasets. The T/R columns represent the terms per rule and Acc represents the accuracy. It is clear from the results that the Ant-miner performs better on the metrics of predictive accuracy as well as simplicity of the rules discovered

[illegible]

DATA SET	Ripper			J48(Decision Tree)			ACO Algorithm		
	Rule	T/R	Acc	Rule	T/R	Acc	Rule	T/R	Acc
Tic_Tac_Toe	9	3	94%	95	1.49	77%	5	1.4	98%
Vote	4	2	94%	6	1.8	96%	2	1.1	97%
Mushroom	9	2	99%	25	1.2	97%	6	1.2	100%

across all the data sets. In terms of complexity of the discovered rules, the rules discovered by Ant Miner are much simpler (smaller) than the rule lists discovered by Ripper classifier. In many data mining applications the simplicity of a rule list/set tends to be even more important than its predictive accuracy and the Ant-Miner seems particularly advantageous as it discovers

Table 2.Experimental Results of Data Sets using ACO algorithm

simpler rules enhancing comprehensibility of the discovered knowledge.

V. CONCLUSION AND FUTUTE WORK

This paper has proposed an algorithm for classification rules discovery called ACOMiner. It is based on the ACO algorithm, a swarm intelligence algorithm, and the Ant-Miner algorithm, an algorithm for classification rules mining. In ACO-Miner algorithm, multi population parallel strategy is proposed, the cost-based discretization method is adopted, and parameters in the algorithm are adjusted step by step. With these improvements, performance of the algorithm is advanced, and predictive accuracy is enhanced than Ant-Miner. Four data sets taken from UCI Repository on Machine Learning have been used in our experiments. The results illuminate that the algorithm proposed in this paper has better performance in predictive accuracy and simplicity of rules.

This paper has implemented an algorithm for rule discovery called Ant-Miner (Ant Colony-based Data Miner) without rule pruning step. The algorithm is based both on the behavior of real ant colonies and on data mining concepts. The performance of the ACO algorithm is validated on three datasets. The experimental results on datasets show that the proposed algorithm discovers classification rules with significantly better accuracy and simplicity as compared to more conventional data mining algorithms like Ripper and J48(decision tree). The proposed algorithm needs to be validated on more number of datasets and it also requires a comparison with later versions of Ant-Miners.

REFERENCES

[1] H. A. Edelstein, Introduction to Data Mining and Knowledge Discovery, Third Edition. Two Crows Corporation, 1999.

[2] J. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques. Elsevier, 2011.

[3] D. Martens, M. De Backer, R. Haesen, J. Vanthienen, M. Snoeck, and B. Baesens, “Classification With Ant Colony Optimization,” Evolutionary Computation, IEEE Transactions on, vol. 11, no. 5, pp. 651 –665, Oct. 2007.

[4] M. Dorigo, V. Maniezzo, and A. Colorni, “Ant system: optimization by a colony of cooperating agents,” Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, vol. 26, no. 1, pp. 29–41, 1996.

[5] Abraham, Ajith; Grosan"Swarm Intelligence in Data Mining,"studies in computational intelligence, vol.34,2006

[6] E. Bonabeau, M. Dorigo, and G. Theraulaz, Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, 1999.

[7] B. Liu, H. A. Abbas, and B. McKay, “Classification rule discovery with ant colony optimization,” in Intelligent Agent Technology, 2003. IAT 2003. IEEE/WIC International Conference on, 2003, pp. 83 – 88.

[8] M. Dorigo and G. Di Caro, “Ant colony optimization: a new meta-heuristic,” in Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on, 1999, vol. 2.

[9] J. Smaldon and A. Freitas, “A new version of the ant-miner algorithm discovering unordered rule sets,” in GECCO, 2006, pp. 43–50.

[10] F. E. B. Otero, A. A. Freitas, and C. G. Johnson, “cAnt-Miner: An Ant Colony Classification Algorithm to Cope with Continuous Attributes,” in Ant Colony Optimization and Swarm Intelligence, vol. 5217, M. Dorigo, M. Birattari, C. Blum, M. Clerc, T. Stützle, and A. F. T. Winfield, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 48–59.

[11] K. Salama and A. Abdelbar, “Extensions to the Ant-Miner Classification Rule Discovery Algorithm,” in Swarm Intelligence, vol. 6234, M. Dorigo, M. Birattari, G. Di Caro, R. Doursat, A. Engelbrecht, D. Floreano, L. Gambardella, R. Groß, E. Sahin, H. Sayama, and T. Stützle, Eds. Springer Berlin / Heidelberg, 2010, pp. 167–178.

[12] H. A. A. B Liu, “Density-based heuristic for rule discovery with ant-miner,” pp. 180–184.

[13] UCI machine learning repositiory databases, http://www.ics.uci.edu/MLRepository.html