Computational Intelligence Techniques for Classification of Cancer Data

Kalagotla Satish Kumar
Gitam University
Visakhapatnam

T. Sita Mahalakshmi, PhD
Gitam University
Visakhapatnam

ABSTRACT

This paper presents improved Ant Colony Optimization (ACO) algorithms for data mining. The goal of the algorithms is to extract classification rules from data. The traditional Ant Colony Optimization algorithm is enhanced with genetic operators to develop improved ACO algorithms. The genetic operators like crossover, mutation are used to develop Ant Colony Optimization with Crossover (ACOC), ACO with mutation (ACOM), and ACO with crossover and mutation (ACOCM). The performance of the improved ACO algorithms is compared with traditional ACO. All the algorithms are applied on three different cancer datasets. The results showed that ACO with mutation gave good accuracy when compared with ACO, ACOC, and ACOCM.

General Terms

Classification, Data Mining.

Keywords

Data Mining, Ant Colony Optimization, Classification, Crossover, Mutation.

1. INTRODUCTION

Data mining is an intersection of machine learning, statistics, swarm intelligence, and data bases. Data mining algorithms are applied in various fields. The core idea in mining data is to discover relevant information, Today's world is changing rapidly with development in technology, As the technology is advancing the information retrieval from large data bases is essential. Data mining algorithms are used to retrieve the relevant information from large data bases.

There are several data mining tasks, including classification, regression, clustering, dependence modeling etc. Each of these tasks is considered as a kind of problem to be solved by data mining algorithm. In this paper, improved ACO algorithm for the classification tasks is presented.

Ant Colony Optimization was introduced by Dorigo et. al. [1][2][3]. Ant colonies are biologically inspired algorithms, The scenario of ant colonies is presented here, ants move randomly in the environment in search of food, ants are blind they lay pheromone on their path in order to communicate with each other by sensing the concentration of pheromone on their trails, every ant move individually and also all ants communicate with each other in their colony, the ant followed the shortest path will reach the food source soon and in return the ants follow the same path which results in the increase of pheromone concentration eventually after a period of time all ants follow the same shortest path which has high accumulation of pheromone.

2. LITERATURE SURVEY

An ant miner algorithm for classification rule extraction is presented and compared with CN2 algorithm by Parpinelli et. al.[4]. Ganji and Abadeh [5] presented an ant colony based classification system to extract a set of fuzzy rules for diagnosis of diabetes disease and its results were compared. Tabakhi and Moradi [6] presented an Unsupervised and multivariate filter-based feature selection methods by analyzing the relevance and redundancy of features. In the methods, the search space is represented as a graph and then the Ant Colony Optimization is used to rank the features. In addition, a heuristic information measure is used to improve the accuracy of the methods by considering the similarity between subsets of features. The performance of the methods was compared to the well-known uni-variate and multivariate methods using different classifiers. Liu et. al. [7] presented Ant miner 3 an improved version of ant miner is applied for classification tasks. Cordon et. al. [8] presented some of the algorithms that developed under ACO framework, an overview of current applications and analyze the relationship between ACO and some of the best known heuristics were presented. Martens et. al. [9] presented an overview of ant based approaches to the classification task and a new ant based classification technique Ant Miner+ is proposed.

3. METHODOLOGY

3.1 Ant Colony Optimization

An Ant Colony Optimization algorithm is designed from the concept of ant colonies. Ants move from nest to food source, they move individually by laying pheromone in the environment. The information among the ants is shared by sensing the pheromone in their path. The coordination among the ant's helps in finding the shortest path from nest to food source. ACO is robust and versatile; it is applied successfully on wide range of combinatorial optimization problems.

ACO algorithm framework is based on the following assumptions.

1) Path followed by an ant from nest to food source is a candidate solution for a given combinatorial optimization problem.
2) The amount of pheromone accumulated on the path is directly proportional to the quality of the candidate solution.
3) The probability of choosing the path depends on the amount of pheromone deposited on the path. The more pheromone deposits more likely the path is chosen.

As a result ants converge to the optimal or near optimal solution for the target problem.

Design of ACO framework

1) An appropriate representation of a problem.
2) Defining a method to construct a valid solution.
3) Defining a problem dependent heuristic function (η) that defines the quality of terms that are to be added to the current partial rule.

4) A rule for pheromone updating, which tells how to modify the pheromone trail (τ).

5) A probabilistic transition rule based on the value of heuristic function (η) and on the contents of the pheromone trails (τ) that is used to iteratively construct a solution.

Transformation of ant behavior into real time application

1) Real ant environment is transformed into a matrix representation for application purpose.
2) Probability of choosing path by ant depends on the pheromone concentration, similarly, probability of choosing a node depends on the pheromone value in application of ant colony concept.
3) Larger pheromone concentration leads to shortest path, similarly high pheromone value on the edges of nodes lead to optimal solution for a target problem.
4) Ants indirectly communicate based on pheromone deposit on each path, similarly, based on comparison of pheromone value on the edges of the nodes next node is selected.

The remnant of this section consist of Description of the proposed algorithm, Construction of a rule, Improved ACO algorithm, rule pruning, Pheromone updating, Pheromone evaporation and the use of discovered rules for classifying new data, Crossover, Mutation.

3.2 Description of the Algorithm

In ACO algorithm each ant gives a candidate solution for the problem. Here, the problem is identification of classification rules. The rule has the form

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

Each term is a triple <attribute, operator, value> value is the value belonging to the domain of the attribute, operator element in the triple is a relational operator. As the attribute element takes nominal values of data the operator element can be either '<=' or '>'.

A high level description of the algorithm is shown in algorithm-I. Ant miner follows a sequential covering approach to discover a list of classification rules covering all or almost all the training cases. Each iteration of while loop corresponding to each iteration of the REPEAT- UNTILL loop, discovers a classification rule. The discovered rules are added to the Discovered rule list, the training cases that are correctly classified by the rules in the discovered rule list are removed from training set and the process is repeated until the training samples are greater than maximum uncovered cases.

Each iteration of REPEAT UNTILL loop consists of four steps they are construction of a rule, rule pruning, pheromone updating and pheromone evaporation.

3.3 Construction of a Rule

Ant_k initially starts with a empty rule, and add term by term to the current partial rule. The choice of term to be added to the current partial rule depends on the problem dependent heuristic function (η) and the pheromone (τ) associated with each term. Ant_k keeps adding one term at a time to its current partial rule until the stopping criteria is met.

a) The term is added to the partial rule if and only if the true positive count of partial rule is less than the true positive count when new term added to the partial rule.

b) If there are no attributes left.

For each candidate solution given by Ant_k, Pheromone (τ) value is updated for the terms considered in the construction of rule R_k. Pheromone evaporation (decrease of pheromone) is updated for the attributes that are not visited by $\text{Ant}_{k.}$ The others ants start constructing the rules by utilizing the updated pheromone values. This process is repeated until the stopping condition is met.

a) The number of rules constructed is greater than the user specified threshold No_of_ants.

Once the REPEAT UNTILL loop is completed the best rules among all the rules constructed by ants is chosen based on the user specified threshold of minimum rule selection after sorting the pheromone values in decreasing order. The best rules are added to the DRL (Discovered Rule List).

The algorithm is executed for different versions of ACO. If the ant_version is 1 then Ant Colony Optimization with crossover is performed. If the ant_version is 2 then Ant Colony Optimization with crossover and mutation is performed. If the ant_version is 3 then Ant Colony Optimization with mutation is performed. If the ant_version is other than the available options then traditional Ant Colony Optimization is performed.

Once REPEAT UNTILL loop and switch operation is performed then the training samples Ts contains only those tuples that are not covered by the rules in the DRL. While loop start with the new training samples, at each iteration of while initialize the same amount of pheromone.

In Algorithm-I, while loop corresponds to different population for rule extraction, from the training set the population is chosen by considering the each attribute values that has repeated more than once. For every attribute the size of repeated values differs, the maximum size of repeated values of the attribute is fixed for all the attributes to make uniform size matrix for calculation purpose by filling the vacant spaces with INF symbol. Values Whereas REPEAT UNTILL loop constitutes the core operation of the algorithm which adds term by term to the partial rule for the construction of rule. Let term_{ij} be a rule condition of the form $A_i = V_{ij}$ where A_i is i th attribute, V_{ij} is the j th value of the domain of ith attribute. The probability that the term_{ij} is chosen to for the current partial rule is

$$P_{ij} = \begin{cases} i = \text{randsample}(n); \Big| \text{if} \sum_{i=1}^{n} \sum_{j=1}^{m} \tau_{ij} = c; \\ j = \text{randi}(m_i) \\ \text{Max}(\sum_{i=1}^{n} \sum_{j=1}^{m} \tau_{ij}); \Big| \text{otherwise}; \end{cases} \quad (1)$$

Where

n is the total number of attributes.

m is the number of domain values of the ith attribute

i is the index of the attribute

j is the domain value index of the ith attribute

c is the first pheromone value $c = \tau_{11}$; first attribute first domain value of first attribute.

Randsample () is the function used to select the random attribute.

Randi () is the function used to select domain value of ith attribute.

Max () is the function used to select the maximum pheromone value.

3.4 Improved ACO Algorithm

Algorithm-I: Improved ACO algorithm

```
{
Ts={} ;    /*Ts=training set*/
DRL= [];  /*DRL: discovered rule list*/
    Ant_version= {1, 2, 3};    /* 1 for ACO with crossover; 2
                                  for ACO with crossover and
                                  mutation; 3 for ACO with
                                  mutation. */
While (Ts>max_uncovered_cases)
K=1;       /*ant index*/
Initialize population;
Initialize pheromone;
Repeat
Ant_k follows a sequential approach by incrementally adding term to the current rule R_k;
Pheromone update at the nodes followed by Ant_k;
Pheromone evaporation is done at the unvisited nodes by Ant_k;
K=K+1;
Until (K>=no_of_ants)
Prune each rule constructed R_k constructed by all ants.
Choose the best rule R_best among all the rules R_k constructed by all ants;
Add R_best to the DRL;
Switch (ant_version)
        Case 1:   Crossover;
                  Add rules to DRL;
                  break;
        Case 2:   Crossover;
                  Mutation;
                  Add rules to DRL;
                  break;
        Case 3:   Mutation;
                  Add rules to DRL;
                  break;
        Default:  break;
End switch
Ts=Ts-{set of cases correctly classified by R_best in DRL};
End while
}
```

3.5 Rule Pruning

The basic idea behind the rule pruning is to improve the quality of the rule. Once the rule is generated the accuracy of the rule R_k is calculated. For each rule R_k, term pruning is performed. After pruning if the rule R_k predicts more accurately then full rule then the pruned term is discarded from the rule or else it is included in the rule. This process is performed for each attribute of the rules in the DRL. The stopping condition for the rule pruning is if the rule is has only one term in it or if all the attributes in the rule are considered for pruning and left with no attribute.

The accuracy of the rule is measured using true positive count. True positive means class1 labeled as class1.

$$\eta = TP \tag{2}$$

3.6 Pheromone Updating

For each trail the selected path is updated with the increase in pheromone. Pheromone update rule is given by

$$\tau_{ij} \leftarrow \tau_{ij} + Q \tag{3}$$

τ_{ij} Represent the pheromone on the edge ij.

Q is the positive constant.

3.7 Pheromone Evaporation

At the end of the each trail pheromone evaporation is done on the unvisited nodes.

$$\tau_{ij} \leftarrow (1-\rho) * \tau_{ij} \tag{4}$$

ρ Represent the evaporation constant.

τ_{ij} Represent the pheromone on the edge ij.

3.8 Use of Discovered Rules

Discovered rule list (DRL) contain the extracted rules, the rules are applied on unknown test cases, for each test case tuple the DRL is applied if any one of the rule classifies the tuple then it is labeled with the rule consequent, and move to next tuple in the test case. If the DRL does not classify the tuple, then the tuple is assigned with a class label of a rule in the DRL that classifies the majority class.

3.9 Crossover

Cross over is usually performed on two parents to generate new strings hoping that will improve the predictive accuracy. Depending on the rule length cross sites are selected. If a rule had nine attributes then the possible cross sites will be eight i.e., length of attributes minus one. Based on the cross site the cross cut is applied, it splits the rule in to two off springs. Similarly the random cross cut is made on the other rule. By combining one end of the cut in first rule and another end of the cut in second rule new rule is generated with the same attribute structure. If the rule has the accuracy greater than the cumulative accuracy of the two rules then the rule is said to be survived or else the rule is not survived. The generated new rules may survive or may not survive. This operation is performed on the rules in DRL.

3.10 Mutation

Mutation is usually performed on the rules to improve its predictive power. Mutation is the small change in the value of a term in a rule. The change may be slight increase or decrease in the value to improve the accuracy of rule. For every attribute in a rule mutation is performed. If there is an improvement in accuracy the mutated value is considered.

The stopping condition is if there are no attributes left to test.

4. DATA SET DESCRIPTION

For this work, the original Wisconsin Breast Cancer (WBC), Mammographic mass data and Haberman datasets are considered. These were taken from UCI Machine Learning Repository [10], to distinguish malignant from benign cases. There were 699 instances with nine attributes and one class label field in total, after removing missing values in the dataset, 683 instances with 444 benign and 239 malignant were considered for the experiment. The problem is to predict a tissue sample taken from a patient's breast is malignant or benign.

Wisconsin Breast Cancer data set has nine integer type attributes namely clump thickness, uniformity of cell size, uniformity of cell shape, marginal adhesion, single epithelial cell size, bare nuclei, bland chromatin, normal nucleoli, mitoses and a class label contains benign and malignant.

In clump thickness the normal cells are grouped in mono layers, where as the cancerous cells are tend to be grouped in multi layers. Uniformity of cell size and shape of cancer cells vary in size and shape, hence variation of cell size and shape plays a crucial role in detecting cancerous cells. In Marginal adhesion the normal cells are bounded where as the cancerous cells lose their ability, hence loss of adhesion results in malignant. Single epithelial cell size is related to the uniformity of cell size, Epithelial cells are significantly enlarged may be a malignant cell. Bare Nuclei is the term used for nuclei that is not surrounded by cytoplasm, this nuclei is typically seen in benign tumours. Bland chromatin describes the uniform texture of the nucleus seen in benign cells; Chromatin of cancerous cells tends to be coarser. The normal nucleoli are the small structures seen in nucleus. In normal nucleoli the nucleus is very small if visible, in cancerous cell the nucleoli are more prominent. Mitosis is the process in which the cell divides and replicates. Pathologists can determine the grade of cancer by identifying mitoses count.

Mammography Mass Data (MMD) is used to predict the severity of a Mammography mass lesion from BI-RADS attribute and patient's age. It contains five attributes namely BI-RADS, age, shape, margin, density and a class label contain either benign or malignant. The data samples considered for training are 558 out of which 286 belong to benign and 272 belong to malignant. Test data contain 279 samples out of which 143 belong to benign and 136 belong to malignant.

Haberman data set contain the cases of survival of the patients after breast cancer surgery. It consists of three attributes and a class label. The attributes are Age of patient at the time of operation, patients year of operation, number of positive axillary nodes, the class label contain the patients who survived at least five years after surgery and the patients who died within five years after surgery.

5. K-FOLD CROSS VALIDATION

K fold cross validation is one of the best measure of performance classifier. The Data set is divided in to each class labeled group. To avoid class imbalance the data tuples in each group are taken as multiples of three. Each group data is distributed in K folds equally. In order to test all samples the K folds are iterated K times. In each iteration, K^{th} fold is taken as test set and (K-1) folds are taken as training set. Mammographic mass data consist of 961 samples, when data is pre processed the noisy data is removed, out of 840 tuples 431 belong to class1 and 409 belongs to class2. 279 belong to test set and 558 belong to training set. Similarly after pre processing Wisconsin breast cancer data consist of 681 samples of which 454 belong to training set and 227 belong to test set. Haberman data consist of 306 samples of which 102 belong to test and 204 belong to training set.

5.1 Result Analysis

When compared with ACO the improved algorithms ACO with mutation and ACO with crossover and mutation gave almost similar results on Wisconsin breast cancer data. ACO with mutation gave best result. Table 1 presents the comparison of algorithms on WBC.

Table 1. Comparison of algorithms on WBC

Folds (WBC)	**ACO**	**ACOC**	**ACOCM**	**ACOM**
1	94.7	92.5	95.6	96
2	93.8	95.6	95.6	95.6
3	95.2	95.6	97.4	97.4
Cumulative Accuracy	**94.6**	**94.6**	**96.2**	**96.3**

When compared with ACO, ACO with mutation gave best results on MMD. Table 2 presents the comparison of algorithms on MMD.

Table 2. Comparison of algorithms on MMD

Folds (MMD)	ACO	ACOC	ACOCM	ACOM
1	83.5	76	81.4	83.5
2	81.7	78.9	81.7	80.3
3	75.3	80.6	81.7	81.7
Cumulative Accuracy	**80.2**	**78.5**	**81.6**	**81.83**

When compared with ACO, ACO with mutation gave better results on Haberman. Table 3 presents the comparison of algorithms on Haberman.

Table3. Comparison of algorithms on Haberman

Folds (Haberman)	ACO	ACOC	ACOCM	ACOM
1	75.5	72.5	76.5	76.5
2	76.5	72.5	77.5	78.4
3	72.5	72.5	74.5	74.5
Cumulative Accuracy	**74.8**	**72.5**	**76.1**	**76.4**

From the above tabulated results it was observed that ACO with mutation showed good accuracy on all the datasets.

5.1.1 Performance Metrics

Performance metrics are the fundamental aspects in datamining to evaluate any technique. Some of the performance metrics for any classifier are Accuracy, Sensitivity, Specificity, Precision, Recall and F-measure. Accuracy is the percentage of correctly labeled tuples to the summation of correctly and incorrectly labeled tuples. Sensitivity is the percentage of truly labeled positives to the summation of positives labeled as positives and positives incorrectly labeled as negatives. Specificity is the percentage of true negatives to the summation of true negatives and false positives. Recall is same as sensitivity, precision is the percentage of correctly labelled positives to the total of positives and negatives labeled as positives. Performance metrics formulae is presented in Table 4. Comparision of performance metrics for each class of WBC, MMD, Haberman are presented in Table 5, Table 6 ,Table 7, Table 8, Table 9and Table 10. In the Tables C1 represent class1 and C2 represent class2.

Table 4. Performance metrics formulae.

Measure	Formula
Accuracy	$\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}$
Sensitivity	$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$
Specificity	$\frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}}$
Precision	$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$
Recall	$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$
F- measure	$F = \frac{2*\mathrm{PRECISION}*\mathrm{RECALL}}{PRECISION + RECALL}$

Where,

TP: The positive tuples that were correctly labeled by the classifier.

TN: The negative tuples that were correctly labeled by the classifier.

FP: The negative tuples that were incorrectly labeled as positives.

FN: The positive tuples that were incorrectly labeled as negatives.

Figure 1 presents the average accuracy of algorithms on WBC, MMD and Haberman.

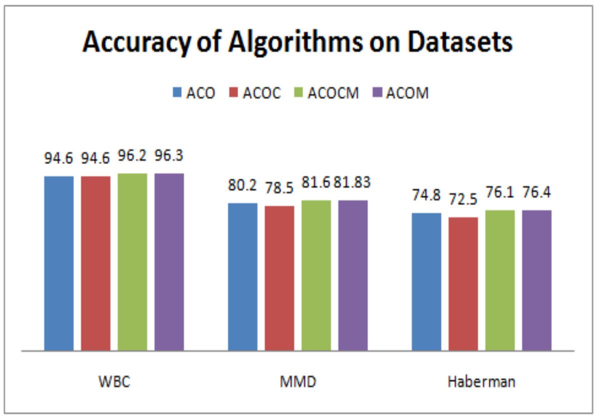

Fig 1: Cumulative accuracy of algorithms on datasets

Table 5. Comparison of class1 performance metrics on WBC

Performance metrics (WBC)	ACO	ACOC	ACOCM	ACOM
	C1	C1	C1	C1
Accuracy	0.9457	0.9457	0.9618	0.9633
Sensitivity	0.9927	0.9857	0.9794	0.9794
Specificity	0.8731	0.8817	0.9306	0.9344
Precision	0.9234	0.9302	0.9617	0.9640
Recall	0.9927	0.9857	0.9794	0.9794
F measure	0.9568	0.9571	0.9705	0.9716

Table 6. Comparison of class2 performance metrics on WBC

Performance metrics (WBC)	ACO	ACOC	ACOCM	ACOM
	C2	C2	C2	C2
Accuracy	0.9457	0.9457	0.9618	0.9633
Sensitivity	0.8731	0.8817	0.9306	0.9344
Specificity	0.9927	0.9857	0.9794	0.9794
Precision	0.9873	0.9747	0.9620	0.9620
Recall	0.8731	0.8817	0.9306	0.9344
F measure	0.9267	0.9259	0.9461	0.9480

Table 7. Comparison of class1 performance metrics on MMD

Performance metrics (MMD)	ACO	ACOC	ACOCM	ACOM
	C1	C1	C1	C1
Accuracy	0.8017	0.7849	0.8160	0.8184
Sensitivity	0.7995	0.7875	0.8161	0.7953
Specificity	0.8040	0.7822	0.8159	0.8478
Precision	0.8182	0.7949	0.8275	0.8695
Recall	0.7995	0.7875	0.8161	0.7953
F measure	0.8088	0.7912	0.8218	0.8307

Table 8. Comparison of class2 performance metrics on MMD

Performance metrics (MMD)	ACO	ACOC	ACOCM	ACOM
	C2	C2	C2	C2

Accuracy	0.8017	0.7849	0.8160	0.8184
Sensitivity	0.8040	0.7822	0.8159	0.8478
Specificity	0.7995	0.7875	0.8161	0.7953
Precision	0.7843	0.7745	0.8039	0.7647
Recall	0.8040	0.7822	0.8159	0.8478
F measure	0.7940	0.7783	0.8099	0.8041

Table 9.Comparison of class1 performance metrics on Haberman

Performance metrics (Haberman)	**ACO**	**ACOC**	**ACOCM**	**ACOM**
	C1	C1	C1	C1
Accuracy	0.7484	0.7255	0.7614	0.7647
Sensitivity	0.8162	0.7527	0.7676	0.7909
Specificity	0.5278	0.4444	0.6818	0.6047
Precision	0.8489	0.9333	0.9689	0.9244
Recall	0.8162	0.7527	0.7676	0.7909
F measure	0.8322	0.8333	0.8566	0.8525

Table 10. Comparison of class2 performance metrics on onHaberman

Performance metrics (Haberman)	**ACO**	**ACOC**	**ACOCM**	**ACOM**
	C2	C2	C2	C2
Accuracy	0.7484	0.7255	0.7614	0.7647
Sensitivity	0.5278	0.4444	0.6818	0.6047
Specificity	0.8162	0.7527	0.7676	0.7909
Precision	0.4691	0.1481	0.1852	0.3210
Recall	0.5278	0.4444	0.6818	0.6047
F measure	0.4967	0.2222	0.2913	0.4194

6. CONCLUSION

In this work the new improved ACO algorithms namely ACO with crossover, ACO with crossover and mutation and ACO with mutation for the effective diagnosis of breast cancer classifier has been proposed. It will help for taking better treatment decisions. The central idea of the proposed model is based on effective rule generation. Traditional ACO generates the rules, when crossover operator is applied on the rules the decrease in accuracy is observed in ACOC when compared with ACO. Similarly, when crossover and mutation operators are applied on the rules generated by ACO then the improvement in accuracy is observed in ACOCM with ACO. In ACOM the generated rules are mutated and the improvement in accuracy is observed when compared to ACO. Early detection of breast cancer is improved with this model. The proposed models, when applied on three clinical datasets proved that ACO with crossover and mutation and ACO with mutation gave almost similar results when compared with traditional ACO. Early detection of breast cancer is improved with this model. The proposed models, when applied on three clinical datasets proved that ACO with crossover and mutation and ACO

ACO with mutation is the best classifier in terms of classification accuracy. This helps in high accurate diagnosis of breast cancer and avoids unnecessary surgery by diagnosis breast masses. The proposed models may gain wider acceptance in the field of breast cancer diagnosis and treatment. Future work aims at classification of clinical datasets with other swarm intelligence techniques like Multi-swarm optimization, Glow warm swarm optimization, The bees algorithm, Artificial bee colony algorithm and Particle swarm optimization algorithm to achieve good classification accuracy.

7. ACKNOWLEDGMENTS

Sincere thanks to TEQIP-II for providing financial assistance for the progress of the work.

8. REFERENCES

[1] Dorigo, M. and Gambardella, L. M. 1997. Ant colony system: a cooperative learning approach to the travelling salesman problem. IEEE transactions on evolutionary computing. 1, 53-66.

[2] Dorigo, M., Maniezzo, V., Colorni, A. The ant system: optimization by a colony of cooperating agents. IEEE Trans Syst Man and Cybernetics-Part B. 26, 1-13.

[3] Gambardella, L, M. and Dorigo, M. 1996. Solving symmetric and asymmetric tsps by ant colonies. IEEE Conference on Evolutionary Computation. 1-6.

[4] Parpinelli, R.S., Lopes, H.S. and Freitas, A. A. 2002. Data mining with an ant colony optimization algorithm. IEEE transactions on evolutionary computing. 6, 321-332.

[5] Ganji, M. F. and Abadeh, M. S. 2011. A fuzzy classification system based on ant colony optimization for diabetes disease diagnosis. Expert systems with applications. 38, 14650-14659.

[6] Tabakhi, S. and Moradi, P. 2015. Relevance-Redundancy feature selection based on ant colony optimization. Pattern recognition. 48, 2798-2811.

[7] Liu, B., Abbass, H. A. and McKay, B. 2004. Classification rule discovery with ant colony optimization. IEEE Computational Intelligence Bulletin, 3, 31-35.

[8] Cordon, O., Herrera, F. and Stutzle, T. 2002. A review on the ant colony optimization metaheuristic: basis, models and new trends. Math ware and Soft Computing. 9, 1-35.

[9] Martens, D., Backer, M. D., Haesen, R., Vanthienen, J., Snoeck, M. and Baesens, B. 2007.Classification with ant colony optimization. IEEE Transactions on evolutionary computing. 11, 651-665.

[10] Lichman, M. Breast Cancer Wisconsin (Original) Dataset, UCI Machine Learning Repository [http://archive.ics.uci.edu/ml] last accessed 19th may 2015.