Received: April 8, 2021. Revised: May 6, 2021. 

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

# Rule Induction with Iterated Local Search

**Ayad Mohammed Jabbar[1]***

[1]*Department of Computer Science, Shatt Alarab University College, Basra, Iraq*
* Corresponding author's Email: AyadMohammed@sa-uc.edu.iq

**Abstract:** The wide amount of data in the modern applications available on the Internet make it very complicated to deal with the knowledge behind these data. The data classification task is a useful tool that used to deal with a huge amount of data by classify these data into coherent groups. The data size decreases the performace of the classification technique, especially when contain uninformative data (i.e., irrelevant, noisy). The stochastic local search algorithm is an optimization approaches employed to find the informative data to build the classification model. These methods are used to search for the optimal patterns to construct the classification algorithm. Thus, this research introduces a stochastic local search algorithm for rule induction called Iterated-Miner (Iterated local search-based rule induction). The purpose of the algorithm is to construct classification rules from data. This classification algorithm is inspired by the concepts and principles of stochastic local search and rule induction. The performance of the proposed classifier evaluated with a well-known and state of art classification algorithms called, Ant-Miner, ACO/PSO2, PART FURIA, and ACO/GA on 10 UCI datasets. The results demonstrate that our classifier is superior compare with all classifiers respect to classification accuracy; and the model sizes by our classifier are considerably competitive with those discovered by other classifier.

**Keywords:** Data mining, Metaheuristic, Rule induction, Rule-based classification, Stochastic local search.

## 1. Introduction

Rule induction is the most widely used machine learning field that creates rules from a set of given database. The inductive learning algorithms are considered as accurate and simple compared to other techniques of machine learning [1, 2]. Inductive learning algorithms are divided into two main groups, which are, rule induction and decision tree induction. The decision tree induction, in which various algorithms have been proposed to convert different decision tree algorithms into a group of rules, such as C4.5, CN2 and ID3 [3, 4]. These algorithms initially create and then transfer a DT to a list of rules by converting each path (i.e., subtree) from the root to the terminal leaf as a classification rule. The second type is known as rule induction algorithm [5], which has the ability to generate if–then rules directly from a given dataset by using various rules-based classifiers, such as conjunctive rule, decision table and one-attribute-rule algorithm.

The rule induction is considered as NP-hard problem. Thus, the optimal rule set cannot be ensured to be reached unless the classifier do a deeper search in the database, which is almost impossible at a reasonable time. However, the conventional rule induction algorithms face local optimazation problems to find the the set of global optima rules [6]. Conventional stochastic local search algorithms such as Tabu Search, Iterated Local Search, Simulated Annealing , and Ant colony optimazation which have been proven effective in settle this type of problems that have fickle search space with different local optima. They perform an exhaustive search via intelligent techniques (e.g., memory, and perturbation) [7-9]. In addition, conventional stochastic local search algorithms are inspired by natural phenomena. The inspiration is materialized in the characteristics and components of these algorithms. These features can be observed by using a population or single point of search. Another significant aspect involves using the memory to guide

the algorithm to discover the search space. Most algorithms employ a single-neighborhood structure which controls the movement in each candidate solution (e.g., Tabu Search, Simulated Annealing, and Ant colony optimization). By contrast, Iterated Local Search utilizes multiple-neighborhood structure search (i.e., local search and perturbation). First, the local search works on each candidate solution until the local optima are reached. Then, the perturbation procedure shifts the search to another point. While doing so, the algorithm exhibits a strong ability to discover the large search space and identify high-quality solutions for the problem at hand. Therefore, this research aims to develop Iterated-Miner, an iterated local search-based rule induction. Iterated-Miner creates rules incrementally, thereby implementing a series of sequential steps to create a rule list that covers as many training instances as possible with the highest classification accuracy. The goal is to ascertain the global optima set of rules to the given datasets by using the concept and principles of the stochastic iterated local search algorithm. Consequently, the Iterated-Miner can be regarded as a hybrid classifier that uses a combination of a perturbation mechanism and subsidiary local search methods.

The remaining of this research is prepared as follows. Section two discusses the iterated local search components and its applications. Section three explained the proposed Iterated-Miner rule induction algorithm. Section four explained the research method used in this research. Section five explained the computational results of the our proposed classifier. The final section highlighted the conclusions part and the future research directions.

## 2. General iterated local search

The importance of metaheuristics algorithm is to deal effectively with the problem of local optima in the iterative improvement methods. Different methods have been considered in the literature when designing a metaheuristic algorithm [10]. Those methods include forbidden areas in the feasible search region (e.g., tabu search), involving the acceptance of worsen solution during the search operations (e.g., simulated annealing), the active principle of populations-based solutions (i.e., evolutionary algorithms), the application of penalties in the search operations as used in the dynamic local search methods, or the concept of swarm intelligent such as used in particle swarm optimization, and ant colony optimization approaches [11].

In like manner, one of the simplest ideas of metaheuristics methods called, iterative improvement local search style. The concept of these methods by producing a new beginning solutions uniformly applied at random from the search areas. Iterated local search (ILS) is stochastic metaheuristics algorithm abel to generate new starting solutions by perturbation of the formerly establish solutions. The basic idea of ILS has been performed in different optimization problem such as travelling salesman problem (TSP) [12], the problem of single machine with total weighted tardiness [13], Single and parallel-machine scheduling [14], Flow shop scheduling [15], Graph bipartitioning [16], MAX-SAT, quadratic assignment problem [17], Job shop scheduling [18], and other optimization problems [19].

The general modular of the ILS framework is determined by four main algorithmic components (i.e., procedures) which they are, (Initial Solution, Perturbation, Local Search, and Acceptance Criterion). The algorithmic component of ILS framework is given in Fig. 1.

The Initial solution procedure, responsible on determines a good beginning stage of the search process. It usually carried out by good constructive heuristic function to produce a better fittness of the initial solutions [16].

The perturbation procedure, responsible to create a new candidate solution from the current established solution. Its task to disturb the under construction solution by producing a new favorable starting one in orderd to escape from local optima. Then, the perturbation solution will be gone under next local search procedure. The perturbation modifications in the solution must be not the same as the modifications that utilized by following local search procedure [20].

The local search is very importance for the ILS performance. The good implementations for this procedure will increased the performance of the solutions that generated by the algorithm. In fact, the concept of local search is to balance between the solutions quality and the time consumption carries to build these solutions. Thus, a fast-local search

Iterated local search

1: $s_0 = \text{GenerateInitialSolution}$
2: $s^* = \text{LocalSearch}(s_0)$
3: **repeat**
4: $s' = \text{Perturbation}(s^*, history)$
5: $s^{*\prime} = \text{LocalSearch}(s')$
6: $s^* = \text{AcceptanceCriterion}(s^*, s^{*\prime}, history)$
7: **until** termination condition met

Figure. 1 Iterated local search (ILS) pseudocode

applied probably will produce less quality solutions (i.e., premature solutions) than other solutions gained by a time-consuming local search method. However, the implementation of local search can problem-dependent and has to be determines based upon experimental results [21].

The Acceptance criterion procedure is also problem-dependent, which aim to discover the quality of the constructed solutions. The balance between both exploration and exploitation can be considered for this procedure either to accept improved solutions only or based on intermediate choices between these extremes (e.g., worse solution can be accepted) . The proposed classifier will be discussed in the next section based on the ILS framework [20].

## 3. The proposed iterated-miner

This section contained a new rule induction classifier called Iterated-Miner, Iterated Local Search-based rul induction. The prime concern of this classifier to find classification rules from the datasets. The metaheuristic algorithm Iterated Local Search (ILS) [20] are used in the process of developing of the classifier. The classification algorithm generates solutions as a classification rule in the form of:

IF < term-1 > And < term-2 > And …< term-N > Then <Class>

Each term in the rule consists of triple of the following <attribute = value>. While the equal "=" operator is always used, preprocessing step must be used to discretize the continuous attributes (real-valued). Furthermore, each attribute in the dataset allowed to use one time in the rule to avoid stumble such as "IF Gender = female) And (Gender = male)". The high-level pseudocode is described in the following algorithm.

The classifier in each iteration performing a sequential process by using the ILS ability to create a list of rules that covering training cases with as possible as high classification accuracy (i.e., optimal rules). The data in the Iterated-Miner classifier represent with integer numbers (e.g., Gender=0, or 1 rather than Gender=male, Gender=female). The graph that represents the data is fully connected. The rule will be discovered while the classifier selects the path from the graph.

The parameters used for the Iterated-Miner classifier are adopted from [20] that introduced the default ILS parameter values. The classifier starts with all training cases in the data. Then, the parameters are initialized and for each iteration of the outer WHILE loop, the classifier is executed if and only if the TainingCases > UncoveredCases, in the same manner the number of implementation of the inner repeat loop is runnig in orderd to construct

Iterated-Miner algorithm 1

```
Input: Data
Output: Rule induction
TainingCases = {all training cases in the data;
RuleInduction List = [ ];
Initialize IterationNo, SimilarityNo, UncoveredCases;
WHILE (TainingCases > UncoveredCases)
IterationIndex =1; SimilarityTest =1;
Repeat
S0 = ConstructRule;
Repeat
    S*=Perturbation (S0);
    S*'=LocalSearch (S*);
    S0=AcceptanceCriterion(S0, S*');
Until termination condition meet
PruneRule();
IF(ConstructRule =PreviousRule)
THEN SimilarityTest = SimilarityTest + 1;
ELSE SimilarityTest = 1;
END IF
IterationIndex = IterationIndex + 1;
UNTIL (IterationIndex >= IterationNo) OR (SimilarityTest >= SimilarityNo)
Add ConstructRule to RuleInduction List;
TainingCases = TainingCases -{CasesCoveredByRule};
END-WHILE
```

Figure. 2 Iterated-Miner algorithm

one classification rule. The UncoveredCases is a user enter parameter of the unclassified training cases so far, usually 10 cases from the dataset.

The classifier in step number nine starts to generate classification rules by using constructive heuristic function (entropy function) to obtain a good quality of the initial classification rules. The heuristic function in our classifier is given by Eqs. (1) and (2) as following:

$$\eta_{ij} = \frac{log_2 C - H(U|A_i = V_{ij})}{\sum_{i=1}^{a} Ci \sum_{j=1}^{bi} (log_2 C - H(U|A_i = V_{ij}))} \quad (1)$$

$$H(U|A_i = V_{ij}) = -\sum_{U=1}^{C} \left[\frac{P(U|A_i = V_{ij})}{T_{ij}}\right] log_2 \left[\frac{P(U|A_i = V_{ij})}{T_{ij}}\right] \quad (2)$$

where

- *U* is the class of the data.
- *C* is the classes number.
- $P(U|A_i = V_{ij})$ is the division consist of the specific cases where the feature data *Ai* has value *Vij* with particular class *w*.
- $|Tij|$ is the full number of cases in partition *Tij* (cases where attribute Ai has specific value *Vij*).
- *a* explain the total number of features.
- *bi* is the number of the possible values in the specific feature i.

The classifier adds one term from the graph at a time to the rule until one of the two conditions appeared:

- Any term from the data inserted to the construction rule will decrease the coverage of the classification rule. Thus, the rule will be covered number of instances smaller than a user specified determination, named, minimum number of cases that covered by the classification rule (NCC).
- The terms number has equally reached to attributes value.

Step number ten to step number fourteen represent the main searching procedures of the Iterated-Miner classifier, these procedures are Perturbation, and Local search. The perturbation task to convert the constructed rule into another rule. It modifies the classification rule and to transform a new, promising starting classification rule for the next local search procedure. The main objective of this procedure is to escape from the collapsing into local optima. This procdure is independent in style from the modifications that been utilized in the local search procedure. The size of the perturbation is measured by the number of changed tems in the constructed rule. This size is very critical, if the modification is too powerful, the rule quality may be lost. The classifier could perform as a random restart search in its behavior. Therefore, the size of the perturbation has been adapted using the basic 2-terms neighborhood. It changes the terms in the rule components by applying random perturbation replacement with other possible terms from its same attribute available in the data.

In step twelve, the discovered rule goes to the local search procedure which is considered one of the most importance for the performance of the classifier. The local search is adopting of 3-Opt technique from travelling salesman problem [22]. The 3-terms removes three terms from the rule and replaces them with other 3 terms from same attributes. This procedure maintains the rule and improves its classification accuracy. Notice that we have used the term/s replacement from the same attribute which would avoid any kind of muddled rule for example "IF (Gender = female) and (Gender = male)". However, to identify the most promising areas in the research, the quality of this rule will be determined by the acceptance criterion procedure.

The acceptance criterion controls the search of the Iterated-Miner classifiers classifier to intensify the search in promising search space regions with high classification quality by Eqs. (3) and (4) adopted from [19].

$$\text{The acceptance criterion} = \begin{cases} S*', \ if \ \text{Quality}\ (S*') > \text{Quality}\ (S0) \\ \\ S0, \ Otherwise \end{cases} \quad (3)$$

$$\text{Quality}\ (S*') = \frac{TP}{TP+FN+FP} + \frac{TN}{FP+TN} \quad (4)$$

where

- *TP* is the total instances number represented by the discovered rule and covered by class classify by the rule.
- *TN* is the instances not covered by the constructed rule and does not covered by the class label classify by the rule.
- *FP* is the instances number not represented by the discovered rule and has the class label predicted by the classification rule.
- *FN* is the instances number represented by the

classification rule and it has class different from the class classify by the rule.

Once the neighborhood search is completed, the rule implements the pruning procedure. The pruning procedure is responsible for excluding terms that decrease its accuracy and adds to the rule during the construction operation. Pruning deletes one term at times while the accuracy of the rule improves, and the quality is computed according to Eq. (4). An example showing the effective of the pruning procedure on a classification rule constructed from a Lymphography dataset before and after pruning is presented below:

The classification rule from Lymphography dataset before pruned rule:

***IF*** block_of_affere = 'yes' ***AND*** changes_in_node = 'lac_margin' ***AND*** no_of_nodes_in = '\(-inf-3.5]\' ***THEN*** 'metastases'

The classification rule from Lymphography dataset after pruned rule:

***IF*** block_of_affere = 'yes' ***THEN*** 'metastases'

The pruning procedure decrease the size terms in the rule, increases it generalization (i.s., the pruning rule covered more training cases), and finally increases it classification accuracy. After the pruning procedure is completed, the current rule is listed to the construction rules list, and the instances covering by the rule are deleted from the training cases. These procedures are iteratively implemented while the number of training cases is greater than uncovered cases. The classifier keeps adding rules to list of discovered rules until one of the following conditions is met:

1- IterationNo reached to the upper limit, the IterationNo is a user-specified threshold.

2- The classifier is convergence to specific rule, the rule constructed rule as exactly same as the previous constructed rule. Therefore, the SimilarityNo is reached to its border, the classifier will then do random restart.

## 4. Research method

In this experiments the well knowing 10-fold cross-validation is used, and the dataset is divided into equally size subsets. In this method, nine subsets will be used in stage of the training, while the remaining one subset will be used for the testing operation. The process of this cross-validation will be repeated 10 times to make sure that all data are utilized for both training and testing stages. Thefere, the average accuracy of all folds is calculated to indicate the performance of the proposed classifier. The 10-fold cross-validation procedure is adapting and adopting as well in other rule induction studies [23].

The implementation of the proposed classifier Iterated-Miner, is evaluated with other five rule induction classifiers, namely, Ant-Miner [24], ACO/PSO2 [25], PART [26] FURIA [27] and ACO/GA [28]. These classifiers are considered the state of art rule induction algorithm. The Ant-Miner introduced by Parpinelli, Lopes and Afreitas (2002) [24] is inspired by the foraging behaviour of a real ant colony. Ant-Miner is a metaheuristic, swarm-based, stochastic and separate-and-conquer approach. This approach has of three main procedure, named, construction rule, pruning rule and updating the pheromone. Ant-Miner has become an effective method for extracting useful and interesting classification rules from data. Artificial ants can find solutions to large and complex search spaces in the context of rule discovery. The strategy of searching space combines exploration and exploitation. The ACO/PSO2 is a hybrid classifier used the search behaviour of metaheuristic swarm intelligence algorithms to find the rule induction model from the data. This classifier used the capabilities of ant colony optimization to deal with discrete attribute, meanwhile it used particle swarm optimization power to cope with continuous features in the data. In this classifier new pruning procedure has been proposed to o delete the unimportant terms from the discovered classification rule. The performance of this classifier was stable, and the results are promising [25]. The PART is a state of the art classification algorithm, firstly, introduced by Frank and Witten (1998). This classifier constructs a partial DT by using C4.5 and converts the subtrees that have the best classification performance into a list of classification rules. These rules represent the discovered patterns in the classification model [26]. The Fuzzy Unordered Rule Induction Algorithm (FURIA) is a modification version of the RIPPER classification algorithm that uses fuzzy logic instead of conventional rules. This classifier has abilities to introduce a classification model that considered a simple and comprehensible. In addition, FURIA employs an active rule-stretching technique to handle uncovered instances from the data. Experimental results display that the accuracy of the FURIA classifier significantly better than the original RIPPER classifier [27]. The ACO/GA is rule based classification algorithm. It used the

Table 1. The datasets description

| Dataset | Description | | |
|---|---|---|---|
| | Instances # | Attributes # | Classes # |
| Balance dataset | 625 | 4 | 3 |
| Ljubljana dataset (Breast Cancer) | 286 | 9 | 2 |
| Wisconsin dataset (Breast Cancer) | 699 | 9 | 2 |
| Dataset of Credit-a | 690 | 15 | 2 |
| Cleveland Heart dataset | 303 | 13 | 5 |
| Statlog Heart dataset | 270 | 13 | 2 |
| Hepatitis dataset | 155 | 19 | 2 |
| Iris dataset | 150 | 4 | 3 |
| Lymphography dataset | 148 | 18 | 4 |
| Sonar dataset | 208 | 60 | 2 |

Table 2. The Iterated-Miner parameters

| Parameter | Description | Value |
|---|---|---|
| IN | Iteration Number | 10 |
| NCC | The number of cases that cover by each rule. | 5 |
| UncoveredCases | The number of uncovered cases by the classification model. | 10 |
| SimilarityNo | Number of rules similarity. | 10 |

combination of two algorithms Ant colony optimization and Genetic algorithm to solve the local optimization problem of Ant-Miner classifier and increase it performance. In this classification algorithm, The ACO responsible for generating candidate rules, while the Genetic algorithm improved these rules by using mutation and crossover operators. Experimental results display this hybridization improve in the classification accuracy [28].

The performance evaluation for this research is carried out using three evaluation criterias. The first one is the correct classification rate which shows the classification performance for the construction classification model. The second criteria is the classification model size that shown by the full amount of terms number in the discovered rule. The number of antecedents (terms) always refer to the number of antecedents taken from the constructed rule. The experiments of this study are implemented by using 10 secondary datasets from UCI repository to test the quality of the proposed classifier [30]. These datasets are widely used for the literature of rule induction studies and explain the different type of feature's numbers, which lie between 4 to 60. The features have two types which they are categorical and continuous types. These 10 datasets has different size of cases number between 148-699 cases. Finally, the class label numbers are also varied. The descriptions of these datasets are displayed on the table below.

The list of parameter values that used in the propsed Iterated-Miner classifier are listed in Table 2.

## 5. Computational result and discussion

The results of the implemented Iterated-Miner classifier are compared with those of four different state of art rule induction classifiers, namely Ant-Miner, ACO/PSO2, PART, FURIA, and ACO/GA. These classification algorithms are considered to be the most-related classification algorithms in the literature. The classification performance of the five classifieres is analysed in detail by using 10 datasets from UCI. Several benchmark scenarios are used for analysing the Iterated-Miner performance. In the first and second scenario's, Tables 3 and 4 show the experimental results of the average classification accuracy, and model size, respectively, by using the 10-fold cross-validation procedure. In each table, the result shows the classification accuracy and, the model size. Meanwhile, the symbol next the result "+/–"are present the number of standard deviations. The best result for each particular data is written in bold text. The results in Tables 3 and 4 are used to determine the best classifiers.

Table 3 shows that the Iterated-Miner outperforms the Ant-Miner, ACO/PSO2, and ACO/GA classifiers in all datasets in terms of classification accuracy, which is the relation between the numbers of instances classified correctly with the dataset size. The Iterated-Miner also achieves 7 out of 10 datasets compared with PART, and FURIA classifiers. The experimental results show that the Iterated-Miner obtains the best classification performance in 6 datasets compared with the all classifiers. Furthermore, the Iterated-Miner obtains the second best results in four datasets, namely Breast Cancer (Wisconsin), Heart (Cleveland), Hepatitis, and Sonar. However, the second best classifiers are PART and FURIA, which acquires the best results in three datasets.

Table 4 demonstrates that the Iterated-Miner obtained the best classification performance in model size for 9 datasets compared with the PART and ACO/GA classifiers. In the same fashion, the Iterated-Miner obtains the best results in 8, and 7 datasets

Table 3. The classifier results based on average classification accuracy

| Dataset | Ant-Miner | ACO/PSO 2 | PART | FURIA | Iterated-Miner | ACO/GA |
|---|---|---|---|---|---|---|
| Balance dataset | 69.73% +/- 1.58% | 68.66 ± 4.97 | 69.92 ± 0.25 | 70.24 % +/- 0.37 | **71.04% +/- 1.22%** | 71.00%+/- 2.31% |
| Ljubljana dataset (Breast Cancer) | 72.32% +/- 1.73% | 70.94 ± 5.37 | 71.32 ± 0.36 | 73.076 % +/- 0.49 | **74.32% +/- 2.87%** | 73.06%+/- 2.01% |
| Wisconsin dataset (Breast Cancer) | 94.43% +/- 1.17% | 93.86 ± 4.56 | 94.70 ± 0.06 | **96.28 % +/- 0.19** | 95.57% +/- 0.62% | 95.50%+/- 0.80% |
| Dataset of Credit-a | 84.49% +/- 1.04% | 84.69 ± 4.39 | 86.37 ± 0.19 | 86.52 % +/- 0.36 | **87.1% +/- 1.49%** | 86.52%+/- 1.22% |
| Cleveland Heart dataset | 76.17% +/- 2.85% | 78.51 ± 6.16 | 78.8 ± 0.09 | **83.1683 % +/- 0.24** | 81.93% +/- 2.44% | 81.34%+/- 2.1% |
| Statlog Heart dataset | 77.78% +/- 2.41% | 78.89 ± 7.78 | **81.85 ± 0.23** | **81.85 % +/- 0.39** | **81.85% +/- 2.17%** | 81.48%+/- 1.56% |
| Hepatitis dataset | 80.03% +/- 3.68% | 76.13 ± 8.34 | **82.58 ± 0.21** | 79.35 % +/- 0.4 | 80.21% +/- 3.29% | 79.75 % +/-0.4 |
| Iris dataset | 94% +/- 1.85% | 94.0 ± 8.14 | 95.33 ± 0.04 | 95.33 % +/- 0.18 | **96% +/- 1.47%** | **96%+/- 1.09%** |
| Lymphography dataset | 71.37% +/- 1.87% | 77.19 ± 12.59 | 79.72 ± 0.11 | 79.05 % +/- 0.2 | **80.36% +/- 5.12%** | 80.26%+/- 3.03% |
| Sonar dataset | 75.61% +/- 2.64% | 54.86 ± 3.87 | **81.25 ± 0.19** | 77.884 % +/- 0.4 | 80.23% +/- 2.66% | 77.80 % +/- 1.2 |

Table 4. The classifier results based on average classification model size

| Dataset | Ant-Miner | ACO/PSO 2 | PART | FURIA | Iterated-Miner | ACO/GA |
|---|---|---|---|---|---|---|
| Balance dataset | 11 +/- 0 | 52 ± 0 | **7.4 ± 1.77** | 24 | 8.9 +/- 0.38 | 13.4+/-0.16 |
| Ljubljana dataset (Breast Cancer) | **7.8 +/- 0.29** | 26.8 ± 6.196 | 38.5 ± 12.9 | 11 | 12.5 +/- 0.31 | 16.1+/-0.81 |
| Wisconsin dataset (Breast Cancer) | **8.4 +/- 0.22** | 17.1 ± 2.42 | 13.6 ± 2.67 | 55 | 10.1 +/- 0.6 | 10.2+/-0.7 |
| Dataset of Credit-a | **10.6 +/- 0.4** | 70.6 ± 7.6 | 41.9 ± 8.79 | 12 | 22.8 +/- 0.8 | 26.5+/-2.11 |
| Cleveland Heart dataset | **9.6 +/- 0.69** | 28.3 ± 4.347 | 33.8 ± 9.48 | 29 | 19.2 +/- 1.13 | 20.5+/-1.34 |
| Statlog Heart dataset | **9.6 +/- 0.58** | 25.9 ± 4.30 | 30.3 ± 9.68 | 23 | 19.8 +/- 1.33 | 18.4+/-1.59 |
| Hepatitis dataset | **8.1 +/- 0.48** | 11.6 ± 2.31 | 14.9 ± 3.31 | 31 | 14.5 +/- 0.93 | 16.2 +/- 1.22 |
| Iris dataset | 3.4 +/- 0.27 | **3.3 ± 0.94** | 3.7 ± 0.94 | 5 | 3.6 +/- 0.43 | 3.8+/-0.44 |
| Lymphography dataset | **9.1 +/- 0.5** | 42.8 ± 6.48 | 26.2 ± 3.96 | 25 | 16.3 +/- 0.26 | 17.7+/-0.87 |
| Sonar dataset | 10 +/- 0.49 | **0.9 ± 1.97** | 31.9 ± 5.95 | 26 | 16.4 +/- 0.45 | 19.2 +/- 1.96 |

compared to FURIA and ACO/PSO2 classifiers respectively. The Ant-Miner gains over 9 datasets compared with Iterated-Miner.

Table 5 and Fig. 3 show the evaluation of the performance based on the nonparametric Holm's post-hoc with Friedman test. This test aims to find the dominant classifier among the 10 datasets. As shown in Table 5, in all cases, the low rank shows the best performance. Thus, Iterated-Miner obtains the best average rank in classification accuracy, and the second best average rank in model size. Fig. 3 indicates the results of the model size rank average versus the classification accuracy rank. The Iterated-Miner classifier achieves the best classifier that balance between the two criterias (i.e., classification accuracy and model siz). Under these circumstances, the Iterated-Miner dominates the other classifiers when considered the both criteria. Thus, the balance between accuracy and comprehensibility is clearly shown in our results: the most accurate results needs to determine the optimal size of classification model to outperform the other classifiers.

Table 5. Rank the classification results based on the performance of all datasets

| Ant-Miner | ACO/PSO 2 | PART | FURIA | Iterated-Miner | ACO/GA |
|---|---|---|---|---|---|
| 5.15 | 5.55 | 3.15 | 2.7 | **1.55** | 2.9 |
| **1.4** | 4.1 | 4.7 | 4.5 | 2.6 | 3.7 |

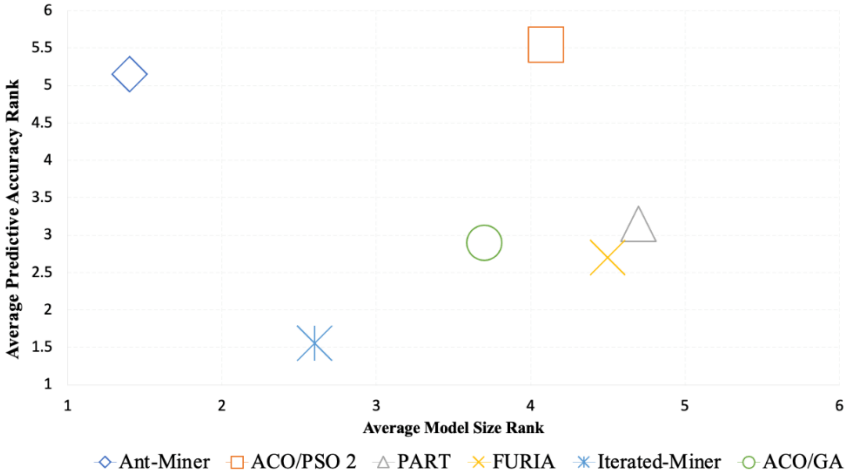


Figure. 3 The classification results based on accuracy rank vs. model size rank

## 6. Conclusion and future work

This research has introduced rules-based classification called Iterated-Miner. The classifier is based on concept and principles of iterated local search, metaheuristic, an optimization algorithms. Extensive experimentation has been conducted on a newly proposed classification algorithm with five states-of-the-art classification algorithms, namely, Ant-Miner, ACO/PSO2, PART, FURIA, and ACO/GA. Our classifier called the Iterated-Miner outperformed the other classifiers in terms of classification accuracy. Despite its convenience as regards classification accuracy, the Iterated-Miner classifier may not achieve the best result to model complexity. The proposed classifier obtained competitive and often better results than the other classifiers. Furthermore, comparison of the Iterated-Miner to a lower model size (i.e., Ant-Miner) made available by Parpinelli et al. (2002) [29] revealed that the former can find all the optimal classification models that are as small as possible and have the highest classification accuracies. The performance evaluation is also conducted with another statistical metric for evaluation, specifically the nonparametric Holm's post-hoc with the Friedman test. This statistical test aims to identify the dominant classifier among all datasets. Thus, the results indicated that the Iterated-Miner classifier acquires the best results that show the balance between the two criteria (i.e., classification accuracy and model size). As future challenges in Iterated local search based rule classifications are: to cope with continuous attributes, the current version of Iterated-Miner is not able to deal with continuous types of attributes and required a pre-processing step. The hybridization with other swarm optimization algorithms such as PSO algorithm is considered an open research direction. Another research direction is to adapt the method that used in evolutionary algorithms to control the size of the perturbation factor to control variation during rule production operation. Finally, the Iterated-Miner classifier could be applied to real-world classification problems such as medical diagnosis and fraud detection.

## Conflicts of Interest

The author declares no conflict of interest.

## Author Contributions

For this paper all contributions (e.g., coding, implementation, result , preparation) have been done by the main author ("Ayad").

## Acknowledgements

The author would like to t express his thanks to Shatt Alarab University College, for supporting this manuscript financially.

## References

[1] H. Elgibreen and M. Aksoy, "Rules - TL: A simple and improved rules algorithm for

incomplete and large data", *J. Theor. Appl. Inf. Technol.*, Vol. 47, No. 1, pp. 28-40, 2013.

[2] H. N. K. Al-behadili, R. Sagban, and K. R. Ku-Mahamud, "Adaptive Parameter Control Strategy for Ant-Miner Classification Algorithm", *Indones. J. Electr. Eng. Informatics*, Vol. 8, No. 1, pp. 149-162, 2020.

[3] A. AlMana and M. Aksoy, "An Overview of Inductive Learning Algorithms", *Int. J. Comput. Appl.*, Vol. 88, No. 4, pp. 20-28, 2014.

[4] D. Farid, L. Zhang, C. Rahman, M.. Hossain, and R. Strachan, "Hybrid decision tree and naïve Bayes classifiers for multi-class classification tasks", *Expert Syst. Appl.*, Vol. 41, No. 4 PART 2, pp. 1937-1946, 2014.

[5] H. N. K. Al-behadili, K. R. Ku-Mahamud, and R. Sagban, "Annealing strategy for an enhance rule pruning technique in ACO-based rule classification", *Indones. J. Electr. Eng. Comput. Sci.*, Vol. 16, No. 3, pp. 1499-1507, 2019.

[6] I. Chorbev, D. Mihajlov, and I. Jolevski, "Web based medical expert system with a self training heuristic rule induction algorithm", *In: Proc. of 2009 1st International Conference on Advances in Databases, Knowledge, and Data Applications, DBKDA*, pp. 143-148, 2009,

[7] H. Hoos and T. Stutzle, "Stochastic Local Search Algorithms: An Overview", in *Springer Handbook of Computational Intelligence*, W. (Eds. Kacprzyk, Janusz, Pedrycz, Ed. Springer, pp. 1085-1105, 2015.

[8] A. M. Jabbar, K. R. Ku-Mahamud, and R. Sagban, "An improved ACS algorithm for data clustering", *Indones. J. Electr. Eng. Comput. Sci.*, Vol. 17, No. 3, pp. 1506-1515, 2020.

[9] A. M. Jabbar, K. R. Ku-Mahamud, and R. Sagban, "Modified ACS Centroid Memory for Data Clustering", *J. Comput. Sci.*, Vol. 15, No. 10, pp. 1439-1449, 2019.

[10] T. Dokeroglu, E. Sevinc, T. Kucukyilmaz, and A. Cosar, "A survey on new generation metaheuristic algorithms", *Comput. Ind. Eng.*, Vol. 137, No. May, pp. 1-29, 2019.

[11] K. Ng, C. Lee, F. Chan, and Y. Lv, "Review on meta-heuristics approaches for airside operation research", *Appl. Soft Comput. J.*, Vol. 66, No. April 2020, pp. 104-133, 2018.

[12] C. Archetti, D. Feillet, A. Mor, and M. Speranza, "An iterated local search for the Traveling Salesman Problem with release dates and completion time minimization", *Comput. Oper. Res.*, Vol. 98, No. August 2020, pp. 24-37, 2019.

[13] E. Queiroga, R. Pinheiro, Q. Christ, A. Subramanian, and A. Pessoa, "Iterated local search for single machine total weighted tardiness batch scheduling", *J. Heuristics*, No. November, pp. 24-37, 2020.

[14] H. Santos, T. Toffolo, C. Silva, and G. Vanden, "Analysis of stochastic local search methods for the unrelated parallel machine scheduling problem", *Int. Trans. Oper. Res.*, Vol. 26, No. 2, pp. 707-724, 2019.

[15] J. Xu, C. Wu, Y. Yin, and W. Lin, "An iterated local search for the multi-objective permutation flowshop scheduling problem with sequence-dependent setup times", *Appl. Soft Comput. J.*, Vol. 52, pp. 39-47, 2017.

[16] L. Helena, O. Martin, and T. Stutzle, "Iterated Local Search: Framework and Applications", in *Handbook of Metaheuristics*, Second Edi., Vol. 146, M. Gendreau and J.-Y. Potvin, Eds. California: Springer US, pp. 363-399, 2019.

[17] S. Shah, "Implementation of iterative local search (ILS) for the quadratic assignment problem", *IEEE Access*, pp. 1-4, 2014.

[18] S. Schulz, J. Neufeld, and U. Buscher, "A multi-objective iterated local search algorithm for comprehensive energy-aware hybrid flow shop scheduling", *J. Clean. Prod.*, Vol. 224, No. 1, pp. 421-434, 2019.

[19] H. N. K. Al-behadili, K. R. Ku-Mahamud, and R. Sagban, "Hybrid Ant Colony Optimization and Iterated Local Search for Rules-Based Classification", *J. Theor. Appl. Inf. Technol.*, Vol. 98, No. 04, pp. 657-671, 2020.

[20] T. Stützle and R. Ruiz, "Iterated Local Search", in *Handbook of Heuristics*, Springer International Publishing, pp. 1-27, 2017.

[21] D. Cattaruzza, N. Absi, D. Feillet, and D. Vigo, "An iterated local search for the multi-commodity multi-trip vehicle routing problem with time windows", *Comput. Oper. Res.*, Vol. 51, No. 1, pp. 257-267, 2014.

[22] J. Bentley, "Fast algorithms for geometric traveling salesman problems", *ORSA journal on computing*, Vol. 4, No. 4. pp. 387-411, 1992.

[23] H. N. K. Al-behadili, K. R. Ku-Mahamud, and R. Sagban, "Hybrid ant colony optimization and genetic algorithm for rule induction", *J. Comput. Sci.*, Vol. 16, No. 7, pp. 1019-1028, 2020.

[24] R. Parpinelli, H. Lopes, and A. Freitas, "Data mining with an ant colony optimization algorithm", *IEEE Trans. Evol. Comput.*, Vol. 6, No. 4, pp. 321-332, 2002.

[25] N. Holden and A. Freitas, "A Hybrid PSO/ACO Algorithm for Discovering Classification Rules in Data Mining", *J. Artif. Evol. Appl.*, Vol. 2008, pp. 1-11, 2008.

[26] R. Datta and S. Saha, "Applying rule-based classification techniques to medical databases : an empirical study", *Int. J. Bus. Intell. Syst. Eng.*, Vol. 1, No. 1, pp. 32-48, 2016.

[27] J. Hühn and E. Hüllermeier, "FURIA: An algorithm for unordered fuzzy rule induction", *Data Min. Knowl. Discov.*, Vol. 19, No. 3, pp. 293-319, 2009.

[28] H. N. K. Al-behadili, K. R. Ku-Mahamud, and R. Sagban, "Hybrid Ant Colony Optimization and Genetic Algorithm for Rule Induction", *J. Comput. Sci.*, Vol. 16, No. 7, pp. 1019-1028, 2020.

[29] R. Parpinelli, H. Lopes, and A. AFreitas, "Data mining with an ant colony optimization algorithm", *IEEE Trans. Evol. Comput.*, Vol. 6, No. 4, pp. 321-332, 2002.

[30] D. Dua and C. Graff, "UCI Machine Learning R epository", *Irvine, CA: University of California, School of Information and Computer Science*, 2 019. [Online]. Available: http://archive.ics.uci.e du/ml.