Computers, Materials & Continua
DOI:10.32604/cmc.2020.013576
Article

Tech Science Press

IWD-Miner: A Novel Metaheuristic Algorithm for Medical Data Classification

Sarab AlMuhaideb*, Reem BinGhannam, Nourah Alhelal, Shatha Alduheshi, Fatimah Alkhamees and Raghad Alsuhaibani

Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh, 11362, Saudi Arabia
*Corresponding Author: Sarab AlMuhaideb. Email: salmuhaideb@ksu.edu.sa
Received: 11 August 2020; Accepted: 19 September 2020

Abstract: Medical data classification (MDC) refers to the application of classification methods on medical datasets. This work focuses on applying a classification task to medical datasets related to specific diseases in order to predict the associated diagnosis or prognosis. To gain experts' trust, the prediction and the reasoning behind it are equally important. Accordingly, we confine our research to learn rule-based models because they are transparent and comprehensible. One approach to MDC involves the use of metaheuristic (MH) algorithms. Here we report on the development and testing of a novel MH algorithm: IWD-Miner. This algorithm can be viewed as a fusion of Intelligent Water Drops (IWDs) and AntMiner+. It was subjected to a four-stage sensitivity analysis to optimize its performance. For this purpose, 21 publicly available medical datasets were used from the Machine Learning Repository at the University of California Irvine. Interestingly, there were only limited differences in performance between IWD-Miner variants which is suggestive of its robustness. Finally, using the same 21 datasets, we compared the performance of the optimized IWD-Miner against two extant algorithms, AntMiner+ and J48. The experiments showed that both rival algorithms are considered comparable in the effectiveness to IWD-Miner, as confirmed by the Wilcoxon nonparametric statistical test. Results suggest that IWD-Miner is more efficient than AntMiner+ as measured by the average number of fitness evaluations to a solution (1,386,621.30 *vs*. 2,827,283.88 fitness evaluations, respectively). J48 exhibited higher accuracy on average than IWD-Miner (79.58 *vs*. 73.65, respectively) but produced larger models (32.82 leaves *vs*. 8.38 terms, respectively).

Keywords: Ant colony optimization; AntMiner+; IWDs; IWD-Miner; J48; medical data classification; metaheuristic algorithms; swarm intelligence

1 Introduction

The medical field is evolving rapidly and it is difficult to overstate its actual and potential importance to human life and societal functioning [1,2]. The focus of this paper is classification problems that can be used to support clinical decisions, such as diagnosis and prognosis. A diagnosis involves providing the cause and

nature of a disorder, and a prognosis predicts a disorder's course [3]. Medical data sets contain collections of patients' recorded histories vis-à-vis diagnoses and/or prognoses. Predictor features for these diagnoses and/or prognoses are given as a set of independent attributes describing a patient case (e.g., demographic data, pathology tests and lab tests). The instances that correspond to attributes, which tend to be continuous or discrete values of a fixed length, are formally given as as $D = \{(x_i, y_i),\ i = 1, \ldots,\ N_{ex}\}$, where N_{ex} is the number of examples, x_i is a vector of the predictor features used to predict y_i and $y_i \in \{1, \ldots,\ N_{cl}\}$, N_{cl} is the number of all possible discrete classes.

The goal of this study is to predict the unknown target class (output) by approximating the relationship between the predictor features (input) and the class.

A central issue in medical data classification is the communication of the acquired knowledge to medical practitioners in a transparent and understandable manner. The model obtained will be used to support a decision made by a human. The acceptance of machine learning models presented for the purpose of medical diagnosis or prognosis is highly dependent on its ability to be interpreted and validated [4]. The use of transparent comprehensible models also allows discovering new interesting relations, and opens new research trends, particularly in the biomedical field [5]. In some cases, obtaining explanations and conclusions that enlighten and convince medical experts is more important than suggesting a particular class. Based on that, Lavrač et al. [6] distinguish between two types of machine learning algorithms according to the interpretability of the acquired knowledge. Symbolic machine learning algorithms, like rule learners and classification trees, can clearly justify or explain their decision through their knowledge representation. On the other hand, subsymbolic machine learning algorithms place more emphasis on other factors, like the predictive accuracy for example. These methods are sometimes called black box methods because they produce their decision with no clear explanation. Examples of this category include artificial neural networks including deep learning methods [7–9], support vector machines [10] and instance-based learners [11]. Statistical learning methods like Bayesian classifiers [12] provide a probability that an instance belongs to some class based on a probability model rather than providing a concrete classification. Learning requires some background knowledge about initial probabilities, and is computationally heavy especially if the network layout is previously unknown.

This work respects the importance associated with generating transparent models that can be easily understood by humans, relating to the syntactic representation of the model as well as the model size. Thus the main requirements of this work are summarized as follows. First, the difficulties associated with medical datasets must be expected, namely the existence of heterogeneous data types, multiple classes, noise, and missing values. Second, model comprehensibility is a central requirement. This includes (a) Knowledge representation of the model must be transparent, and in a form that is easily interpreted by the users, and (b) The size of the obtained knowledge representation must be manageable. Smaller sizes are preferred to larger ones, given a comparable predictive accuracy. Third, the proposed classification model must have a comparable predictive accuracy to current benchmark machine learning classification algorithms, in order to gain acceptance and credibility.

The aim is to achieve this goal by restricting our attention to rule-based models because they are transparent and comprehensible from the perspective of users. More specifically, we confine our attention to metaheuristic methods because they offer three key advantages: fast problem solving, amenability to relatively large problem contexts and the generation of algorithms which are multi-dimensionally robust [13–15]. There are two kinds of metaheuristic algorithms: single-solution based and population-based. The former chooses one solution to improve and mutate during the search lifetime e.g., simulated annealing [16,17] and variable neighborhood search [18,19]. By contrast, population-based algorithms improve the entire population. Evolutionary algorithms [20] and swarm intelligence [21,22] are examples in this respect.

We focus on developing and testing a novel algorithm based on swarm intelligence which fuses aspects of two extant algorithms. Accordingly, the remainder of the paper is structured as follows. In Section 2 we summaries related work in this domain of research. Next, methodological details pertaining to the development of IWD-Miner are presented in Section 3. In Section 4, the algorithm is subjected to a four-stage sensitivity analysis for the purpose of performance optimization. In Section 5, we compare the performance of the optimized IWD-Miner against AntMiner+ and J48. Finally, brief conclusions including limitations and suggestions for future research are provided in Section 6.

2 Related Work

2.1 Swarm Intelligence Algorithms

Swarm intelligence (SI) methods are inspired by the collective behavior of agents within a swarm. SI behaviors have two properties: Self-organization and division of labor. Thus, they focus on the relationships between individuals' cooperative behaviors and their prevailing environment [23–25]. Self-organization can produce structures and behaviors for the purpose of task execution [26,27]. Since there are different tasks carried out by specialized individuals concurrently inside the swarm, a division of labor must be applied, where the cooperating individuals are executing concurrent tasks in a more efficient way than if unspecialized individuals executed sequential tasks [28,29]. The occurrence of indirect actions, such as cooperative behavior and communication between individuals inside a limited environment, is called stigmergy [30,31].

Relying on the behaviors of social animals and insects like ants, birds and bees, SI algorithms aim to design systems of multiple intelligent agents [32]. Examples of imitated social behaviors are a flock of birds looking to discover a place that has enough food and ants finding an appropriate food source within the shortest distance.

2.2 Intelligent Water Drops

Many SI algorithms have been developed, one example being Intelligent Water Drops (IWDs) which was first put forward by Shah-Hosseini [33] and then elaborated on by that author in subsequent publications [34–36]. As its name suggest, this algorithm mimics what happens to water drops in a river [37]. The river is characterized by its water flow velocity and its soil bed, both of which affect the IWDs [35]. Initially, an IWD's velocity will equal a default value and the soil will be initiated as zero before subsequently changing while navigating the environment. There is an inverse relationship between velocity and soil. Thus, a path with less soil produces more velocity for an IWD [37]. The time taken is directly proportional to the velocity and inversely proportional to the distance between two locations [35].

An (N, E) graph represents the environment of IWD algorithms, where E denotes the set of edges associated with the amount of soil and N reflects the set of nodes. Each IWD will travel between N through E and thus gradually construct its solution. The total-best solution T^{TB} depends on the amount of soil included within the edges of that solution. The algorithm terminates once the expected solution is found or once the maximum number of iterations $iter_{max}$ is reached.

2.3 Ant Colony Optimization

Other examples of SI algorithms are framed in terms of ant colony optimization (ACO). Individual ants have limited capabilities, such as restricted visual and auditory communication. However, a group of ants, together in a colony, has impressive capabilities, such as foraging for food. That type of natural optimization behavior achieves a balance between food intake and energy expenditure. When ants find a food source, each ant has the ability to produce chemicals (called pheromones) to communicate with the

rest of the ants. Ants search for their own path that will lead from the source (nest) to the destination (food source). The path will be labelled with pheromones and is thus followed by other ants.

The first person to leverage the cooperative behavior of ants for producing an algorithm was Marco Dorigo in his doctoral research [38]. His work has since been developed and extended by many scholars e.g., [39–42].

Drawing on earlier work by Dorigo and others, Parpinelli et al. [43] proposed the Ant-Miner algorithm for classification problems wherein the classification rules are represented as IF $(term_1 \wedge term_2 \wedge \ldots \wedge term_n)$, THEN *class*, where n is the number of possible terms chosen. Only categorical attributes are operationalised in this version of Ant-Miner and the algorithm is divided into three phases: rule construction, rule pruning and pheromone updating. First, in the iterations of the rule construction phase, the ant will start to search for terms for constructing the rule. Each term will be added to the partial rule based on the probability $P_{i,j}$ of choosing $term_{i,j}$, the heuristic value $\eta_{i,j}$ of $term_{i,j}$, which, according to information theory, is measured by entropy, and the pheromone trail value $\tau_{i,j}$. In this step, if the constructed rule has irrelevant terms, it will immediately be excluded, which is important for enhancing the quality of the rule. Assigning a pheromone value occurs in the pheromone updating phase; that is, in the first iteration, each trial of $term_{i,j}$will be given the same pheromone value, which is small when the number of attribute values is high, and vice versa. Then, deposited pheromone values will be increased if $term_{i,j}$ is added to the rule, and decreased for pheromone evaporation.

Since the original publication of Ant-Miner, scholars have developed and explored variants. For example, Martens et al. [44] proposed a technique using a modified version of Ant-Miner classification methods based on the MAX-MIN Ant System algorithm [41]. This new version is called AntMiner+. In this variant, a directed acyclic graph is used in the rule construction phase and only the best ant performs the pheromone updating and rule pruning phases. Reference [45] proposed another variant which they called mAnt-Miner+ wherein multi-populations of ants are considered i.e. an ant colony divided into several populations. The populations run separately with the same number of ants. After all ants have constructed rules, the best ant is selected to update pheromones, thus minimising the cost of computation compared to the case where all ants can update pheromones.

3 Methodology

3.1 IWD-Miner

In what follows we detail the development of a novel algorithm which can be viewed as a fusion of IWDs and AntMiner+. More specifically, the skeleton of the algorithm is as per the latter, but with ants replaced by IWDs and pheromones replaced by soil.

3.1.1 Initialization

A directed acyclic construction graph is configured following Martens et al. [44]. This is based on the predictor features in the input datafile. Initial parameter values are as per Tab. 1. The graph is initialized with an equal amount of soil on all edges (1000). There are 1000 IWDs and each begins its trip with zero soil $(soil^{IWD})$. The velocity of each IWD is 4 based on [34]. Optimal velocity is then determined experimentally. While travelling, each IWD constructs its solution, gathers soil and gains velocity.

3.1.2 Move Operator

IWDs move through the construction graph commencing from the *Start* vertex towards the *End* vertex forming a rule as follows. From the current value (node) in the current vertex group (attribute), the IWD probabilistically chooses a value (node) from the next vertex group (attribute) according to the heuristic used, the velocity of the IWD and the soil available on graph edges (Eq. (1), which is also a function of

Eq. (2), where $v_c(IWD)$is the list of nodes visited by this IWD, $\epsilon_s = 0.001$ and $g(soil(i,j))$ is used to shift $soil(i,j)$ on the path joining nodes i and j).

Table 1: Initial parameter values

Parameters	Values
In*itSoil* (*IS*)	1000
Vel^{IWD}	4
n*bIWD*	1000
ρ_{IWD}, ρ_n	−0.9, 0.9

$$p^{IWD}(j) = \frac{f(soil(i,j))^{\alpha} \times HUD^{\beta}}{\sum_{k \notin v_c(IWD)} f(soil(i,k))^{\alpha} \times HUD^{\beta}} \tag{1}$$

$$f(soil(i,j)) = \frac{1}{\varepsilon_s + g(soil(i,j))} \tag{2}$$

A heuristic then measures the undesirability of a move by an IWD from i to j (Eq. (3)). Next, the IWD's velocity (Eq. (4)) and soil (Eq. (5), which is also a function of Eq. (6)) are updated. The path with less soil will have a higher chance of being selected and the solution is constructed in a sequential coverage manner, where the data points covered by the rule are removed from the dataset as per the AntMiner+ approach [44].

$$HUD = \frac{\left|T_{ij} \notin Class_{IWD}\right|}{\left|T_{ij}\right|} \tag{3}$$

where T_{ij} reflects the set of remaining training instances that contain $term_{ij}$, $Class_{IWD}$ represents the class that the IWD contributes to for rule extraction, $a_v = 1$, $c_v = 1$ and $b_v = 0.01$.

$$Vel^{IWD}(t+1) = Vel\ (t) + \frac{a_v}{b_v + c_v \cdot soil^2(i,j)} \tag{4}$$

$$soil^{IWD} = soil^{IWD} + \Delta soil(i,j) \tag{5}$$

$$\Delta soil(i,j) = \frac{1}{\varepsilon + \frac{HUD(j)}{Vel^{IWD}(t+1)}} \tag{6}$$

3.1.3 Objective Function

To evaluate the complete rule (solution) represented by an IWD, we follow the AntMiner+ approach [44] and use both rule confidence and rule coverage as fitness as in Eq. (7). Rule confidence is a measure of how many instances containing this specific condition belong to this class at the same time. It measures the reliability of a rule. On the other hand, rule coverage concerns how many instances in this class contain this specific condition over the total instances.

$$Q = \frac{\text{freq}(\text{rule}_{\text{IWD}}, \text{class}_{\text{rule}})}{\text{freq}(\text{rule}_{\text{IWD}})} + \frac{\text{freq}(\text{rule}_{\text{IWD}}, \text{class}_{\text{rule}})}{\text{TD}} \tag{7}$$

where the numerators of confidence and coverage reflect the number of rules correctly classified by this IWD; the denominator of confidence is the total number of instances covered by this IWD's rule ($\text{freq}(\text{rule}_{\text{IWD}})$); and the denominator of coverage is the total instances in the dataset (TD).

3.1.4 Convergence

The convergence process for solving classification problems is inspired by IWDs and AntMiner+. It aims to limit the evaluated IWDs to reach the targeted rules to complete the search faster. This is achieved when all soil in the global best path is arc_{min} shown in Eq. (8) and all soil on the remaining paths is arc_{max} (Eq.(9))

$$\text{arc}_{\text{min}} = \text{iterations}\left(\left(\rho_{\text{IWD}} - \rho_{\text{n}}\ \text{nbIWD}\right)\left(\frac{1}{\varepsilon}\right)\right) \tag{8}$$

$$\text{arc}_{\text{max}} = (\rho_0\ \rho_s)^{\text{iterations}}\text{IS} \tag{9}$$

where nbIWD is the number of IWDs, ρ_n is the local soil updating parameter, ρ_{IWD} is the global soil updating parameter, ρ_0 is the complement of ρ_n, ρ_s is the complement of ρ_{IWD} and IS represents the initialization of the soil.

3.1.5 Termination

Termination depends on many factors, including reaching the minimum convergence threshold, which represents the allowable number of convergence rules (equals 10). Termination also occurs if no water drops find a rule to cover any data instance.

3.1.6 Pseudocode

As shown in Algorithm 1, the graph is constructed in Line 1. Then, there are two main cycles: outer and inner. The initialization of the soil and the heuristic occurs on the remaining datapoints in the training set in each outer cycle in Lines 3 and 4.

Algorithm 1: Pseudocode for IWD-Miner

```
1. ConstructGraph();
2. while NotTerminationCondition() do
3.         InitilizeSoil();
4.       InitializeHeuristic();
5.         while NotConverged() do
6.               IWDs = CreateNewWaterDrops();
7.               i =  Start point
8.             ConstructSolution(IWDs);
9.               T^TB = FindBestIWD(T^IWD)
10.                PruneRule(T^TB)
```

Algorithm 1 (continued).

```
11.                 UpdatePathSoil(T^TB);
12.             end
13. rule = extractRule();
14. RemoveFromTrainingset(CoveredByRule);
15. end
```

In the inner cycle, the IWDs are created in Line 6. They begin their trip at a *Start* point and construct a solution (Line 8 of Algorithm 1 and elaborated on in Algorithm 2). This involves choosing a value from the next variable starting with α, β, *Class* and then the other variables following Eqs. (1)–(3) (Line 4 in Algorithm 2). Next, each IWD updates its velocity (Line 5 in Algorithm 2 using Eq. (4)) and soil (Line 6 in Algorithm 2 using Eq. (5) and Eq. (6)). When all IWDs are complete, the best solution is selected based on its fitness in Line 9 according to Eq. (7).

Algorithm 2: ConstructSolution(IWDs)

```
1. while NotMaxNumberOfIWDs() do
2.   InitializeVelocity();
3.   while IWDsNotConstructSolutions () do
4.         j = ChooseNextNode(IWDs);
5.         UpdateVelocity(IWDs);
6.         Soil^IWD = Soil^IWD + ComputeSoil(i,j);
7.         i = j;
8.         UpdateSoil(i,j);
9.   end
10.  SaveRule();
11. end
```

3.2 Data and Pre-Processing

For the purposes of optimizing IWD-Miner and comparing its performance with two rival algorithms, 21 medical datasets are used (Tab. 2). These are all publicly available from the Machine Learning Repository at the Department of Information and Computer Science University of California, Irvine (UCI). *Inst*. is the number of instances, *Attr*. is the number of attributes, *Num*. are numeric attributes, *Nom*. are nominal attributes, and *Class* refers to the number of classes. The percentage of overall missing values (%*MV*) was computed as $\frac{\#missing\ values}{(Inst.\ \times\ Attr.)} \times 100$, the percentage of instances with missing values (%*Inst.MV*) was computed as $\frac{\text{Inst. with missing values}}{\text{Inst.}} \times 100$. Finally, *IR* is the class imbalance ratio which can be applied if the dataset contains a binary class imbalance or multi-class imbalance; there is no universally agreed threshold for the class imbalance ratio but sometimes $IR > 1.5$ is used [46].

Table 2: Overview of the UCI datasets

No.	Dataset	*Inst.*	*Attr.*	*Num.*	*Nom.*	*Class*	*%MV*	*%Inst.MV*	*IR*
1	Echo	132	12	9	3	2	8.33	100	2.08
2	Wdbc	569	31	30	1	2	0	0	1.68
3	Haber	306	3	3	0	2	0	0	2.77
4	Puba	345	6	6	0	2	0	0	1.37
5	Dia	768	8	8	0	2	0	0	1.86
6	Breast-cancer	286	9	3	6	2	0.34	3.14	2.36
7	Breast-w	699	9	9	0	2	0.25	2.28	1.9
8	Diabetes	768	8	8	0	2	0	0	1.86
9	Heart-c	303	13	6	7	5	0.18	2.31	1.19
10	Heart-statlog	270	13	6	7	2	0	0	1.25
11	Hepatitis	155	19	6	13	2	5.71	48.9	3.84
12	Mammographic_masses	961	5	1	4	2	3.37	16.85	1.15
13	ThoracicSurgery	470	19	3	13	2	0	0	5.71
14	Caesarean	80	5	2	3	2	0	0	1.35
15	Pima_diabetes	768	8	8	0	2	0	0	1.86
16	Liver-disorders	345	6	6	0	2	0	0	1.37
17	Spectfheart	267	44	44	0	2	0	0	3.85
18	Thyroid	7200	21	21	0	3	0	0	40.15
19	Cleveland	303	13	13	0	5	0.18	2.31	12.61
20	Saheart	415	9	8	1	2	0	0	1.88
21	Haberman	306	3	3	0	2	0	0	2.77

Next, we used the Fayyad and Irani Discretizer as per the approach employed in AntMiner+. Finally, we used correlation-based feature subset selection which resulted in 10 features being selected.

3.3 Evaluation

We use standard 10-time, 10-fold cross-validation to evaluate algorithm performance with respect to each of the datasets. Five different performance criteria are used to gauge the relative efficacy of different variants of IWD Miner (Section 4) as well as the relative efficacy of the optimized IWD-Miner compared to two extant, rival algorithms (Section 5).

The first criterion is sensitivity as in Eq. (10), where *TP* represents the number of positive records that have been correctly predicted as positive and *pos* refers to the number of positive records.

$$Sn = \frac{TP}{pos} \tag{10}$$

The second criterion is specificity as in Eq. (11), where *TN* represents the number of positive records that have been correctly predicted as negative and *neg* refers to the number of negative records.

$$Sp = \frac{TN}{neg} \tag{11}$$

The third criterion is accuracy as in Eq. (12), where FP represents the number of negative records that have been incorrectly predicted as positive and FN represents the number of positive records that have been incorrectly predicted as negative:

$$Acc = \frac{TP + TN}{(TP + TN + FP + FN)} \tag{12}$$

The fourth criterion is model size (Eq. (13a) for decision lists and Eq. (13b) for decision trees), where *Rules* denotes the average number of rules per individual, T/R is the average number of terms per rule, and TL is the total number of leaves in a tree. Note that the measuring unit for *Size* is terms, while that for $Size_{J48}$ is leaves.

$$Size = Rules \times T/R \tag{13a}$$

$$Size_{J48} = TL \tag{13b}$$

The fifth and final criterion is efficiency (*AES*) which is measured by the average number of evaluations needed to find a solution in successful runs, undefined when a solution is not found. The measuring unit for AES is fitness evaluations.

4 Sequential Optimization

In what follows IWD-Miner is subjected to a four-stage sensitivity analysis. The best performing model from each stage is taken forward to the next stage in a process of sequential optimization.

4.1 Stage One: Path Update Procedure

First, we compared two models: $\text{IWD} - \text{Miner}_{Best}$ and $\text{IWD} - \text{Miner}_{Each}$ (Tab. 3). The former only updates the soil path of the best IWD (Algorithm 1, Line 11) and as such does not consider the soil update between adjacent vertexes (Algorithm 2, Line 8). The latter updates the soil paths of all IWDs and chooses the best performing model. As such, the path's soil is updated twice: first when each IWD's selected vertex updates the path between the previous vertex and the next vertex (Algorithm 2, Line 8) and second when the path's soil of the best IWD is updated (Algorithm 1, Line 11). In terms of the differential performance of these two models across the 21 datasets, Wilcoxon signed-rank tests did not detect a statistically significant difference between the two models in terms of *Sp* ($p = 0.986$), *Sn* ($p = 0.765$), *Acc* ($p = 0.768$) or *Size* ($p = 0.375$). However, there was a significant difference between the two models as measured by *AES* ($p = 0.001$), where $\text{IWD} - \text{Miner}_{Each}$had the performance advantage. Accordingly, $\text{IWD} - \text{Miner}_{Each}$ is taken through to the next stage.

Table 3: Optimisation: path update procedure

		$\text{IWD} - \text{Miner}_{Each}$	$\text{IWD} - \text{Miner}_{Best}$
Acc	**Avg**	73.11 ± 0.162	**73.22 ± 0.159**
	Min	43.01 ± 0.022	**43.16 ± 0.035**
	Max	95.70 ± 0.006	**95.91 ± 0.006**

(Continued)

Table 3 (continued).

		IWD − Miner$_{Each}$	IWD − Miner$_{Best}$
Sp	**Avg**	**60.21 ± 0.315**	60.06 ± 0.314
	Min	**2.71 ± 0.026**	1.43 ± 0.010
	Max	96.13 ± 0.009	**96.48 ± 0.064**
Sn	**Avg**	**75.25** ± 0.237	74.92 ± 0.236
	Min	20.56 ± 0.008	**20.63** ± 0.008
	Max	98.48 ± 0.008	98.48 ± 0.008
Size	**Avg**	**8.51**	8.66
	Min	0.09	0.16
	Max	**37.99**	40.94
AES	**Avg**	**5,628,922.60**	11,958,791.01
	Min	34,600.10	**1,832.90**
	Max	**85,157,856.00**	89,350,560.00

The superior performance of IWD − Miner$_{Each}$ with respect to *AES* can be understood in the following terms. As IWDs are affected by each other, updating the soil path of each IWD increases the probability of choosing the same path by the next IWD, which means producing a large number of duplicate rules. Hence, a small number of unique rules will be evaluated after the filtering step. Considering performance with respect to the properties of the datasets, IWD − Miner$_{Each}$ had a better AES in most datasets featuring high imbalance ratios (*IR*) such as wdbc, haberman and saheart. Moreover, the best Max *AES* value was associated with the thyroid dataset, which has the largest number of instances and the highest IR. Furthermore, for most datasets with missing values (%MV) and instances with missing values (%Inst.M,) IWD − Miner$_{Each}$ had better AES. Examples include the breast-cancer dataset, mammographic-masses dataset and heart-c dataset. In sum, as $\%MV$, $\%Inst.MV$, the number of instances and *IR* increase so does the relative performance of IWD − Miner$_{Each}$ compared to IWD − Miner$_{Best}$ in terms of the *AES* criterion.

4.2 Stage Two: Global Soil Updating Parameter

Next, we experimented with three different values of the global soil updating parameter ρ_{IWD}: −0.5, −0.7 and −0.9 (Tab. 4). Similar to what was observed above in stage one, these three models performed similarly well across the 21 datasets. Friedman tests did not detect statistically significant differences in performance in terms of *Sp* (p = 0.172), *Sn* (p = 0.538), *Acc* (p = 0.495) or *Size* (p = 0.827). However, there was a significant difference between the three models as measured by *AES* (p = 0.000) with ρ_{IWD} = −0.5 having the performance advantage. Accordingly, IWD − Miner$_{Each}$ (stage one) with ρ_{IWD}= −0.5 (stage two) is taken forward to stage three.

Table 4: Optimization of global soil updating parameter

		$\rho_{IWD} = -0.5$	$\rho_{IWD} = -0.7$	$\rho_{IWD} = -0.9$
Acc	**Avg**	**73.53 ± 0.151**	73.24 ± 0.156	73.11 ± 0.162
	Min	42.99 ± 0.029	42.49 ± 0.006	**43.01 ± 0.022**
	Max	95.98 ± 0.007	**96.08 ± 0.008**	95.70 ± 0.00
Sp	**Avg**	59.77 ± 0.308	60.18 ± 0.322	**60.21 ± 0.315**
	Min	1.57 ± 0.018	1.98 ± 0.027	**2.71 ± 0.026**
	Max	96.11 ± 0.008	**96.25 ± 0.010**	96.13 ± 0.009
Sn	**Avg**	**76.18 ±** 0.219	74.83 ± 0.242	75.25 ± 0.237
	Min	**21.24 ±** 0.010	18.23 ± 0.113	20.56 ± 0.008
	Max	**98.95 ±** 0.005	98.58 ± 0.007	98.48 ± 0.008
Size	**Avg**	8.82	8.90	**8.51**
	Min	0.15	0.07	0.09
	Max	39.32	41.55	**37.99**
AES	**Avg**	**5,026,373.13**	8,441,961.26	5,628,922.60
	Min	**5,762.50**	10,039.30	34,600.10
	Max	**76,647,384.00**	146,220,480.00	85,157,856.00

4.3 Stage Three: Number of IWDs (nbIWD)

Here our purpose is to explore the impact of the number of IWDs (nbIWD) on model performance. We compared the performance of four model variants across the 21 medical datasets. Specifically, we explored the ramifications of 500, 750, 1000, 1250 and 2000 IWDs (Tab. 5). Interestingly, no significant differences in performance were revealed across the five models with respect to any of the performance criteria according to Friedman tests: *Acc* (p = 0.785), *Sp* (p = 0.394), *Sn* (p = 0.938), *Size* (p =0.450) and *AES* (p = 0.349). Accordingly, we retain the default value of 1000 IWDs.

Table 5: Optimisation: number of IWDs

		n$bIWD = 500$	n$bIWD = 750$	n$bIWD = 1000$	n$bIWD = 1250$	n$bIWD = 2000$
Acc	**Avg**	**74.98 ± 0.027**	74.70 ± 0.035	74.57 ± 0.033	74.50 ± 0.051	74.44 ± 0.034
	Min	58.99 ± 0.097	57.97 ± 0.167	**62.09 ± 0.150**	60.13 ± 0.161	56.57 ± 0.140
	Max	**93.61 ± 0.035**	92.13 ± 0.038	87.70 ± 0.035	87.21 ± 0.181	92.46 ± 0.036
Sp	**Avg**	59.93 ± 0.075	**68.23 ± 0.089**	60.32 ± 0.090	64.43 ± 0.096	59.59 ± 0.094

(Continued)

Table 5 (continued).

		n*bIWD* = 500	n*bIWD* = 750	n*bIWD* = 1000	n*bIWD* = 1250	n*bIWD* = 2000
	Min	17.80 ± 0.051	14.68 ± 0.052	21.62 ± 0.039	20.91 ± 0.048	**35.99 ± 0.090**
	Max	94.51 ± 0.067	95.83 ± 0.288	91.07 ± 0.078	**96.10 ± 0.281**	91.07 ± 0.071
Sn	**Avg**	80.61 ± 0.052	**80.61 ± 0.056**	79.66 ± 0.059	79.64 ± 0.083	80.41 ± 0.064
	Min	51.51 ± 0.105	51.83 ± 0.087	**53.71 ± 0.087**	52.65 ± 0.071	49.83 ± 0.095
	Max	95.36 ± 0.018	**95.82 ± 0.019**	94.68 ± 0.013	95.01 ± 0.016	94.44 ± 0.023
Size	**Avg**	7.57	7.85	7.94	**7.56**	7.80
	Min	0.90	0.75	0.84	0.79	0.71
	Max	16.58	18.25	15.22	15.74	**15.06**
AES	**Avg**	885,857.32	**862,568.74**	1,013,528.48	957,618.48	948,579.48
	Min	3,629.90	3,666.50	5,762.50	**2,410.10**	3,441.80
	Max	**1,465,441.20**	1,762,808.20	1,876,326.10	1,542,044.50	1,698,935.80

The revealed insensitivity of the algorithm in this respect can be understood in the following terms. IWD-Miner filters duplicate rules so that only unique rules are left; varying the number of IWDs impacts on the number of rules but not the number of unique rules.

4.4 Stage Four: Initial IWD Velocity (Vel^{IWD})

Finally, we explore the impact of alternative initial IWD velocity (Vel^{IWD}) values. Again, leveraging the 21 medical datasets, we compared the model where $Vel^{IWD} = 4$ with three models where $Vel^{IWD} = 0$, 8 and 10 (Tab. 6). Friedman tests did not detect statistically significant differences between the four models in terms of *Sp* (p = 0.440) and *Size* (p = 0.113). However, significant differences were revealed with respect to the remaining three criteria: *Sn* (p = 0.043), *Acc* (p = 0.035) and *AES* (p= 0.000). The model where Vel^{IWD} = 10 proved to be superior in terms of *AES*, but in terms of Sn and Acc, the model where Vel^{IWD} = 8 performed best. We therefore opted for Vel^{IWD} = 8 in the final version of IWD-Miner.

Table 6: Initial IWD velocity

		$Vel^{IWD} = 0$	$Vel^{IWD} = 4$	$Vel^{IWD} = 8$	$Vel^{IWD} = 10$
Acc	**Avg**	72.50 ± 0.030	73.53 ± 0.032	**73.65 ± 0.028**	71.95 ± 0.038
	Min	44.06 ± 0.036	42.99 ± 0.029	42.90 ± 0.022	**44.26 ± 0.036**
	Max	95.71 ± 0.006	**95.98 ± 0.007**	95.96 ± 0.009	95.80 ± 0.012
Sp	**Avg**	75.36 ± 0.074	76.18 ± 0.075	**76.46 ± 0.064**	74.14 ± 0.081
	Min	21.12 ± 0.013	**21.24 ± 0.010**	19.36 ± 0.085	20.80 ± 0.008

Table 6 (continued).

		$Vel^{IWD} = 0$	$Vel^{IWD} = 4$	$Vel^{IWD} = 8$	$Vel^{IWD} = 10$
	Max	98.60 ± 0.007	**98.95 ± 0.005**	98.33 ± 0.010	97.48 ± 0.034
Sn	**Avg**	60.43 ± 0.070	59.77 ± 0.073	**70.40 ± 0.068**	59.81 ± 0.079
	Min	1.140 ± 0.009	1.570 ± 0.018	3.140 ± 0.023	**3.860 ± 0.036**
	Max	**97.54 ± 0.024**	96.11 ± 0.008	95.96 ± 0.005	95.75 ± 0.009
Size	**Avg**	8.82	8.82	8.38	**7.75**
	Min	0.17	0.15	0.09	0.22
	Max	**36.72**	39.32	40.94	36.74
AES	**Avg**	5,607,029.83	5,026,373.13	4,314,405.43	**2,195,514.68**
	Min	5,634.10	5,762.5	3,972.70	**1,315.00**
	Max	82,980,864.00	76,647,384.00	72,491,256.00	**34,774,200.00**

IWD velocity controls the amount of soil that will be gathered by each IWD as it moves; greater velocity leads the IWD to accumulate more soil by removing it from the path [47]. The amount of soil removed from the riverbed by an IWD during movement will affect the probability of the next IWD choosing the same path, which generates similar rules. Thus, the number of rules will decrease after applying a filtering step to retain only unique rules. This explains why a velocity of 10 proved superior in terms of efficiency but the lower value of 8 had the advantage in terms of other performance criteria.

5 Algorithm Inter-Comparison

We explored the relative performance of the optimized IWD-Miner against AntMiner+ and J48. Default settings were used for both rival algorithms, with an equal number of agents between AntMiner+ and IWD-Miner.

AntMiner+ is closely related to our algorithm and thus it seems prudent to evaluate whether and the extent to which the former can be modified in ways which have performance advantages. J48 was chosen as a comparator because it is a well-known and widely used classification algorithm which produces transparent models. IWDs is not amenable to classification problems in its original form hence its omission from this exercise.

We compared the optimized IWD-Miner with the rival AntMiner+ algorithm in terms of accuracy, efficiency, sensitivity, size of model and specificity. With the exception of efficiency, these same metrics were then used to compare IWD-Miner and J48. Efficiency is not calculable with respect to J48, hence its omission. The detailed results are shown in Tab. 7 and Tab. 8. Wilcoxon signed-rank tests did not detect a statistically significant difference between IWD-Miner and AntMiner+ in terms of *Acc* (p = 0.414), *Sp* (p = 0.244), *Sn* (p = 0.876) or *Size* (p = 0.322). However, a significant difference between the two algorithms was revealed with respect to *AES* (p = 0.010) with IWD-Miner exhibiting superior performance. The same non-parametric test was then used to compare the performance of IWD-Miner and J48. No significant differences were revealed between the two models in terms of *Sn* (p = 0.986) or

Sp ($p = 0.821$). J48 exhibited superior performance in terms of *Acc* ($p = 0.04$) but our proposed algorithm was revealed to have a statistically significant advantage in terms of *Size* ($p = 0.000$). Taking the breast-w dataset as an example, both IWD-Miner and J48 were comparable in terms of accuracy but the size of the former model was much smaller than the latter: 8 *vs*. 27.

Table 7: Performance evaluation: IWD-Miner *vs*. AntMiner+

	IWD-Miner					AntMiner+				
No. dataset	*Acc*	*Sn*	*Sp*	*Size*	*AES*	*Acc*	*Sn*	*Sp*	*Size*	*AES*
1. Echo	93.44 ± 0.042	93.56 ± 0.038	93.36 ± 0.110	1.59	3,476.32	97.05 ± 0.013	98.16 ± 0.018	94.02 ± 0.013	1.32	7,375.47
2. Haber	82.88 ± 0.095	55.00 ± 0.148	88.83 ± 0.071	3.70	12,782.70	83.63 ± 0.159	67.80 ± 0.290	90.83 ± 0.213	4.32	464.44
3. Pima_diabetes	82.90 ± 0.017	77.48 ± 0.026	88.03 ± 0.067	6.17	657,728.76	83.13 ± 0.143	76.05 ± 0.155	89.83 ± 0.247	5.81	425.22
4. Heart-statlog	72.37 ± 0.021	45.59 ± 0.030	86.72 ± 0.068	8.56	406,286.23	73.24 ± 0.025	47.92 ± 0.022	86.79 ± 0.065	12.11	289,787.52
5. Breast-cancer	95.49 ± 0.023	95.90 ± 0.016	94.74 ± 0.081	8.07	69,463.31	95.07 ± 0.011	95.65 ± 0.012	93.96 ± 0.038	6.69	471,026.34
6. Puba	58.25 ± 0.028	78.33 ± 0.081	43.00 ± 0.107	3.19	16,997.05	56.50 ± 0.037	57.33 ± 0.091	56.25 ± 0.128	3.59	371.97
7. Diabetes	76.05 ± 0.019	89.21 ± 0.033	60.55 ± 0.102	13.35	352,380.86	76.55 ± 0.019	89.64 ± 0.033	61.18 ± 0.090	16.22	1,228,244.77
8. Hepatitis	84.15 ± 0.059	98.33 ± 0.131	3.14 ± 0.71	6.52	14,576.17	84.36 ± 0.044	98.80 ± 0.123	1.86 ± 0.045	5.08	51,631.75
9. Mammographic_mass	77.94 ± 0.006	19.36 ± 0.047	93.11 ± 0.42	9.40	379,055.94	77.87 ± 0.007	17.03 ± 0.028	93.64 ± 0.017	8.48	2,074,373.09
10. Dia	93.59 ± 0.021	89.58 ± 0.101	95.96 ± 0.42	11.01	330,767.61	93.92 ± 0.016	90.14 ± 0.083	96.16 ± 0.024	9.89	2,997,439.44
11. Breast-w	95.96 ± 0.005	84.15 ± 0.147	84.15 ± 0.009	40.94	776,982.53	96.20 ± 0.003	74.28 ± 0.006	85.23 ± 0.009	46.58	913,736.94
12. Caesarean	67.61 ± 0.042	83.92 ± 0.051	36.79 ± 0.103	4.39	15,268.43	67.49 ± 0.058	84.76 ± 0.191	34.93 ± 0.167	5.16	45,977.42
13. Heart-c	43.30 ± 0.018	95.78 ± 0.010	5.24 ± 0.049	0.17	255,311.26	44.26 ± 0.015	91.99 ± 0.032	9.65 ± 0.043	0.19	725,816.85
14. ThoracicSurger	73.58 ± 0.008	88.00 ± 0.171	46.69 ± 0.023	9.35	70,491.62	73.40 ± 0.006	88.05 ± 0.007	46.04 ± 0.014	9.54	618,208.01
15. Liver-disorders	42.90 ± 0.022	96.47 ± 0.006	4.04 ± 0.085	0.09	13,894.39	43.42 ± 0.034	95.04 ± 0.110	6.00 ± 0.139	0.15	383.79
16. Spectfheart	78.89 ± 0.014	89.67 ± 0.085	65.42 ± 0.027	13.64	407,312.67	76.59 ± 0.009	90.40 ± 0.135	59.33 ± 0.032	14.23	538,110.99
17. Thyroid	63.89 ± 0.009	53.10 ± 0.096	67.68 ± 0.052	0.89	23,938,706.00	56.93 ± 0.005	61.16 ± 0.055	55.44 ± 0.027	0.71	47,459,802.69
18. Cleveland	64.41 ± 0.011	66.83 ± 0.016	57.97 ± 0.008	0.93	475,528.89	56.14 ± 0.009	55.25 ± 0.015	58.45 ± 0.007	0.69	753,816.09
19. Saheart	73.13 ± 0.018	88.40 ± 0.073	44.62 ± 0.113	9.17	120,840.07	73.95 ± 0.019	88.47 ± 0.062	46.82 ± -.111	8.82	153,030.40

Table 7 (continued).

	IWD-Miner					AntMiner+				
No. dataset	*Acc*	*Sn*	*Sp*	*Size*	*AES*	*Acc*	*Sn*	*Sp*	*Size*	*AES*
20. Haberman	53.10 ± 0.113	21.58 ± 0.125	81.17 ± 0.196	15.24	15,701.24	53.23 ± 0.143	20.89 ± 0.155	80.78 ± 0.274	18.98	425.22
21. Wdbc	72.74 ± 0.008	95.37 ± 0.018	18.07 ± 0.005	8.94	785,495.34	74.12 ± 0.008	95.93 ± 0.017	21.28 ± 0.009	11.82	1,042,513.00
Avg	73.65 ± 0.028	76.46 ± 0.064	59.97 ± 0.068	8.35	1,386,621.30	73.19 ± 0.037	75.46 ± 0.078	60.40 ± 0.018	9.07	2,827,283.88
Min	42.90 ± 0.022	19.36 ± 0.047	3.14 ± 0.071	0.09	3,476.32	43.42 ± 0.034	17.03 ± 0.028	1.86 ± 0.045	0.15	371.97
Max	95.96 ± 0.005	98.33 ± 0.131	95.96 ± 0.042	40.94	23,938,706.00	97.05 ± 0.013	98.80 ± 0.123	96.16 ± 0.024	46.58	47,459,802.69

Table 8: Performance evaluation: IWD-Miner *vs*. J48

	IWD-Miner				J48			
No. Dataset	*Acc*	*Sn*	*Sp*	*Size*	*Acc*	*Sn*	*Sp*	*Size*
1. Echo	93.44 ± 0.042	93.56 ± 0.038	93.36 ± 0.110	1.59	97.30	97.30	94.40	3
2. Haber	82.88 ± 0.095	55.00 ± 0.148	88.83 ± 0.071	0.93	71.90	71.90	44.80	5
3. Pima_diabetes	82.90 ± 0.017	77.48 ± 0.026	88.03 ± 0.067	9.35	73.44	73.40	67.80	45
4. Heart-statlog	72.37 ± 0.021	45.59 ± 0.030	86.72 ± 0.068	14.35	77.04	77.00	76.10	39
5. Breast-cancer	95.49 ± 0.023	95.90 ± 0.016	94.74 ± 0.081	8.94	75.52	75.50	47.60	8
6. Puba	58.25 ± 0.028	78.33 ± 0.081	43.00 ± 0.107	0.17	66.96	67.00	63.50	51
7. Diabetes	76.05 ± 0.019	89.21 ± 0.33	60.55 ± 0.102	9.17	73.44	73.40	67.80	45
8. Hepatitis	84.15 ± 0.059	98.33 ± 0.131	3.14 ± 0.071	3.70	83.87	83.90	54.20	27
9. Mammographic_mass	77.94 ± 0.006	19.36 ± 0.047	93.11 ± 0.042	6.17	85.18	82.50	82.10	11
10. Dia	93.59 ± 0.021	89.58 ± 0.101	95.96 ± 0.042	8.56	73.44	73.40	67.80	45
11. Breast-w	95.96 ± 0.005	84.15 ± 0.006	84.15 ± 0.009	8.07	94.13	94.10	92.80	27
12. Caesarean	67.61 ± 0.042	83.92 ± 0.147	36.79 ± 0.103	3.19	65.00	65.00	63.40	16
13. Heart-c	43.30 ± 0.018	95.78 ± 0.051	5.24 ± 0.049	13.35	77.56	77.60	23.50	59
14. ThoracicSurger	73.5 ± 0.008	88.00 ± 0.101	46.69 ± 0.023	6.52	84.47	84.50	16.00	1
15. Liver-disorders	42.90 ± 0.022	96.47 ± 0.071	4.04 ± 0.085	0.09	66.96	67.00	63.50	51
16. Spectfheart	78.89 ± 0.014	89.67 ± 0.085	65.42 ± 0.027	9.40	73.41	73.40	43.30	43
17. Thyroid	63.89 ± 0.009	53.10 ± 0.096	67.68 ± 0.052	40.94	99.68	99.70	99.00	33
18. Cleveland	64.41 ± 0.011	66.83 ± 0.016	57.97 ± 0.008	15.24	99.68	53.10	23.60	67
19. Saheart	73.13 ± 0.018	88.40 ± 0.073	44.62 ± 0.113	4.39	67.23	67.20	55.20	34
20. Haberman	53.10 ± 0.113	21.58 ± 0.125	81.17 ± 0.196	0.89	71.57	71.60	44.70	5
21. Wdbc	72.74 ± 0.008	95.37 ± 0.018	18.07 ± 0.005	11.01	97.30	93.50	93.10	25

(Continued)

Table 8 (continued).

	IWD-Miner				J48			
No. Dataset	*Acc*	*Sn*	*Sp*	*Size*	*Acc*	*Sn*	*Sp*	*Size*
Avg	73.65 ± 0.028	76.46 ± 0.064	59.97 ± 0.068	0.93	79.58	77.24	61.15	5
Min	42.90 ± 0.022	19.36 ± 0.047	3.14 ± 0.071	9.35	65.00	53.10	16.00	45
Max	95.96 ± 0.005	98.33 ± 0.131	95.96 ± 0.042	14.35	99.68	99.70	99.00	39

The relative accuracy of J48 could be a function of data attributes. For example, using the liver-disorders dataset, which contains 7 attributes, J48 achieved 67% accuracy compared to 43% achieved by IWD-Miner. On the other hand, using the spectfheart dataset, which contains 45 attributes, J48 and IWD-Miner achieved 73% and 93% accuracy, respectively. However, one of the strengths of J48 is the execution time it takes to build the model, which is very small compared to IWD-Miner. Another strength is its ability to handle missing values rather than eliminating records containing missing data, leading to loss of training data.

6 Conclusion

The results of the algorithm inter-comparison exercise testify to the competitiveness of IWD-Miner in comparison to two extant alternatives. IWD-Miner was shown to be as good as AntMiner+ in terms of four out of five performance criteria and better than AntMiner+ with respect to the remaining criterion, efficiency. The performance of our algorithm was comparable to that of J48 in terms of two out of four criteria; the latter was advantageous in terms of accuracy but IWD-Miner was superior in terms of resulting model sizes. As such, assuming these results generalize to other contexts (i.e., other datasets), it appears that IWD-Miner could be successfully used for the purpose of medical data classification tasks. Nevertheless, there is ample scope for future research to extend the current study in terms of algorithm development and testing. IWD-Miner takes a relatively long time to run in its current form. To mitigate this issue, we recommend exploring if and the extent to which the convergence process of the algorithm can be improved. Further, although the sensitivity analysis conducted herein suggested limited differences between IWD-Miner variants, there is scope for exploring alternative methods of discretization and feature subset selection and, finally, it may be prudent to implement a delayed procedure for handling missing values after attribute selection.

Acknowledgement: The authors would like to thank the anonymous reviewers for their constructive comments.

Funding Statement: This research project was supported by a grant from the "Research Center of the Female Scientific and Medical Colleges", the Deanship of Scientific Research, King Saud University.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] K. D. Boudoulas, F. Triposkiadis, C. Stefanadis and H. Boudoulas, "The endlessness evolution of medicine, continuous increase in life expectancy and constant role of the physician," *Hellenic Journal of Cardiology*, vol. 58, no. 5, pp. 322–330, 2017.

[2] P. Conrad, *The medicalization of society: On the transformation of human conditions into treatable disorders*. Baltimore, MD: Johns Hopkins University Press, 2007.

[3] B. Gylys and M. E. Wedding, *Medical terminology systems: A body systems approach*. Philadelphia, PA: Davis Company, 2009.

[4] M. J. Pazzani, S. Mani and W. R. Shankle, "Acceptance of rules generated by machine learning among medical experts," *Methods of Information in Medicine*, vol. 40, no. 5, pp. 380–385, 2001.

[5] I. Inza, B. Calvo, R. Armañanzas, E. Bengoetxea, P. Larrañaga *et al.,* "Machine learning: An indispensable tool in bioinformatics," in *Bioinformatics Methods in Clinical Research,* R. Matthiesen (ed.), Totowa, NJ: Human Press, pp. 25–48, 2010.

[6] N. Lavrač and B. Zupan, "Data mining in medicine," in *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach (ed.), Boston, MA: Springer, pp. 1111–1136, 2010.

[7] M. S. P. Subathra, M. A. Mohammed, M. S. Maashi, B. Garcia-Zapirain, N. J. Sairamya *et al.,* "Detection of focal and non-focal electroencephalogram signals using fast walsh-hadamard transform and artificial neural network," *Sensors*, vol. 20, no. 17, pp. 4952, 2020.

[8] M. A. Mohammed, K. H. Abdulkareem, S. A. Mostafa, M. K. A. Ghani, M. S. Maashi *et al.,* "Voice pathology detection and classification using convolutional neural network model," *Applied Sciences*, vol. 10, no. 11, pp. 3723, 2020.

[9] E. M. Hassib, A. I. El-Desouky, L. M. Labib and E. S. M. El-kenawy, "WOA+ BRNN: An imbalanced big data classification framework using whale optimization and deep neural network," *Soft Computing*, vol. 24, no. 8, pp. 5573–5592, 2020.

[10] J. S. Chou, C. P. Yu, D. N. Truong, B. Susilo, A. Hu *et al.,* "Predicting microbial species in a river based on physicochemical properties by bio-inspired metaheuristic optimized machine learning," *Sustainability*, vol. 11, no. 24, pp. 6889, 2019.

[11] M. K. Abd Ghani, M. A. Mohammed, N. Arunkumar, S. A. Mostafa, D. A. Ibrahim *et al.,* "Decision-level fusion scheme for nasopharyngeal carcinoma identification using machine learning techniques," *Neural Computing and Applications*, vol. 32, no. 3, pp. 625–638, 2020.

[12] T. Li, S. Fong and A. J. Tallón-Ballesteros, "Teng-Yue algorithm: A novel metaheuristic search method for fast cancer classification," in *Proc. of the 2020 Genetic and Evolutionary Computation Conf. Companion*, Cancun, Mexico, pp. 47–48, 2020.

[13] S. AlMuhaideb and M. E. B. Menai, "HColonies: A new hybrid metaheuristic for medical data classification," *Applied Intelligence*, vol. 41, no. 1, pp. 282–298, 2014.

[14] N. Dey, *Advancements in applied metaheuristic computing*. Hershey, PA: IGI Global, 2017.

[15] E. G. Talbi, *Metaheuristics: From design to implementation*. Hoboken, NJ: Wiley, 2009.

[16] E. Aarts, J. Korst and W. Michiels, *Search methodologies*, G. Kendall, Boston, MA: Springer, pp. 646–651, 2005.

[17] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[18] P. Hansen and N. Mladenović, "Variable neighborhood search: Principles and applications," *European Journal of Operational Research*, vol. 130, no. 3, pp. 449–467, 2001.

[19] P. Hansen, N. Mladenović, R. Todosijević and S. Hanafi, "Variable neighborhood search: Basics and variants," *EURO Journal on Computational Optimization*, vol. 5, no. 3, pp. 423–454, 2017.

[20] A. K. Tanwani and M. Farooq, "Classification potential vs. classification accuracy: A comprehensive study of evolutionary algorithms with biomedical datasets, " in *Learning Classifier Systems*, J. Bacardit, W. Browne, J. Drugowitsch, E. Bernadó-Mansilla, M. V. Butz *et al.,* Berlin, Heidelberg: Springer, pp. 127–144, 2009.

[21] A. E. Hassanien and E. Emary, *Swarm intelligence: Principles, advances, and applications*. Boca Raten, FL: CRC Press, 2018.

[22] J. Kennedy, "Swarm intelligence, " in *Handbook of Nature-Inspired and Innovative Computing*, A. Y. Zomaya, Boston, MA: Springer, pp. 187–219, 2006.

[23] J. C. Bansal, P. K. Singh and N. R. Pal, *Evolutionary and swarm intelligence algorithms*. Cham, Switzerland: Springer, 2019.

[24] S. Das, A. Abraham and A. Konar, "Swarm intelligence algorithms in bioinformatics, " in *Computational Intelligence in Bioinformatics*, A. Kelemen, Berlin, Heidelberg: Springer, pp. 113–147, 2008.

[25] R. S. Parpinelli and H. S. Lopes, "New inspirations in swarm intelligence: A survey," *International Journal of Bio-Inspired Computation*, vol. 3, no. 1, pp. 1–16, 2011.

[26] J. L. Deneubourg, S. Aron, S. Goss and J. M. Pasteels, "The self-organizing exploratory pattern of the argentine ant," *Journal of Insect Behavior*, vol. 3, no. 2, pp. 159–168, 1990.

[27] M. Dorigo and T. Stützle, *Ant colony optimization*. Cambridge, MA: The MIT Press, 2004.

[28] D. Karaboga, "An idea based on honey bee swarm for numerical optimization. Turkey: Erciyes University, Technical Report, 2005.

[29] R. Xiao and Y. Wang, "Labour division in swarm intelligence for allocation problems: A survey," *International Journal of Bio-Inspired Computation*, vol. 12, no. 2, pp. 71–86, 2018.

[30] M. Dorigo, E. Bonabeau and G. Theraulaz, "Ant algorithms and stigmergy," *Future Generation Computer Systems*, vol. 16, no. 8, pp. 851–871, 2000.

[31] G. Theraulaz and E. Bonabeau, "A brief history of stigmergy," *Artificial Life*, vol. 5, no. 2, pp. 97–116, 1999.

[32] O. Deepa and A. Senthilkumar, "Swarm intelligence from natural to artificial systems: Ant colony optimization," *International Journal on Applications of Graph Theory in Wireless Ad hoc Networks and Sensor Networks*, vol. 8, no. 1, pp. 9–17, 2016.

[33] H. Shah-Hosseini, "Problem solving by intelligent water drops," in *2007 IEEE Congress on Evolutionary Computation*, Singapore, pp. 3226–3231, 2007.

[34] H. Shah-Hosseini, "Intelligent water drops algorithm," *International Journal of Intelligent Computing and Cybernetics*, vol. 1, no. 8, pp. 193–212, 2008.

[35] H. Shah-Hosseini, "The intelligent water drops algorithm: A nature-inspired swarm-based optimization algorithm," *International Journal of Bio-Inspired Computation*, vol. 1, no. 1–2, pp. 71–79, 2009.

[36] H. Shah-Hosseini, "An approach to continuous optimization by the intelligent water drops algorithm," *Procedia—Social and Behavioral Sciences*, vol. 32, pp. 224–229, 2012.

[37] S. Pothumani and J. Sridhar, "A survey on applications of IWD algorithm," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 3, no. 1, pp. 1–6, 2015.

[38] M. Dorigo, "Optimization Learning and Natural Algorithms," PhD thesis. Politecnico di Milano, Italy, 1992.

[39] B. Bullnheimer, R. F. Hartl and C. Strauß, "A new rank based version of the ant system: A computational study," *Central European Journal for Operations Research and Economics*, vol. 7, pp. 25–38, 1997.

[40] W. J. Gutjahr, "A graph-based ant system and its convergence," *Future Generation Computer Systems*, vol. 16, no. 8, pp. 873–888, 2000.

[41] T. Stutzle and H. Hoos, "MAX-MIN ant system and local search for the traveling salesman problem," in *Proc. of 1997 IEEE Int. Conf. on Evolutionary Computation*, Indianapolis, IN, pp. 309–314, 1997.

[42] E. G. Talbi, O. Roux, C. Fonlupt and D. Robillard, "Parallel ant colonies for combinatorial optimization problems, " in *Parallel and Distributed Processing, IPPS 1999, Proceedings: Lecture Notes in Computer Science (LNCS 1586)*, J. Rolim, F. Mueller, A. Y. Zomaya, F. Ercal, S. Olariu *et al.,* Berlin, Heidelberg, 239–247, 1999.

[43] 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.

[44] D. Martens, M. De Backer, R. Haesen, J. Vanthienen, M. Snoeck *et al.,* "Classification with ant colony optimization," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 5, pp. 651–665, 2007.

[45] H. Wu and K. Sun, "A simple heuristic for classification with ant-miner using a population," in *4th Int. Conf. on Intelligent Human-Machine Systems and Cybernetics*, Nanchang, pp. 239–244, 2012.

[46] M. Lango, "Tackling the problem of class imbalance in multi-class sentiment classification: An experimental study," *Foundations of Computing and Decision Sciences*, vol. 44, no. 2, pp. 151–178, 2019.

[47] G. James, D. Witten, T. Hastie and R. Tibshirani, *An introduction to statistical learning with application in R*. New York, NY: Springer Science+Business Media, 2013.