Transactions on Machine Intelligence 4(3) (2021) 137-152

ISSN Online: 2821-1693

Transactions on Machine Intelligence

Journal Homepage: https://tmachineintelligence.ir/index.php/TMI

Feature Selection Method Based on the Rough Set Theory and the Intelligent Water Drops Algorithm

E. Dalvand [1,*], M. Bafandkar [2]

[1] Department of Electrical and Computer Engineering, Shahid Beheshti University, Tehran, Iran
[2] Department of Computer Engineering, Shiraz Branch, Islamic Azad University, Shiraz, Iran

ARTICLE INFO

Article History:
Received 5 June 2021
Received in revised form 21 August 2021
Accepted 17 September 2021
Available online 18 September 2021

Keywords:
Feature Selection, Rough Sets, Dimensionality Reduction, Intelligent Water Drops, Ant Colony

ABSTRACT

Since datasets are often collected for purposes other than data mining, they typically contain numerous redundant and irrelevant features. The presence of such features can pose challenges for learning systems, including increased computational costs and reduced accuracy. Consequently, feature selection as a preprocessing step can enhance system performance in applications that utilize datasets. This paper presents a method that employs rough set theory as a criterion for evaluating the quality of a feature subset and the intelligent water drops algorithm as a search method. The primary objective of this study is to reduce the computational complexity associated with applying rough set theory in feature selection while improving the quality of the final solution set through the use of the intelligent water drops search algorithm. The proposed method has been tested on multiple datasets from the University of California, Irvine (UCI) repository, and its results have been compared with existing similar methods. The findings demonstrate that eliminating redundant computations can significantly reduce the time complexity of rough set-based feature selection and that the intelligent water drops algorithm serves as an efficient search strategy for feature selection.

1. INTRODUCTION

With advancements in data collection techniques and improvements in computer hardware, large-scale datasets have emerged, making knowledge discovery from these databases a critical research area. Researchers are actively developing methods and tools for machine learning to extract knowledge from collected observations and empirical data. For most of these methods, feature selection is an essential preprocessing step. Feature selection serves two primary purposes: first, it enables data compression, simplifying the understanding of analyzed information; second, it enhances the training process of classifiers [1]. Furthermore, reducing the number of features accelerates the classifier training process.

Feature selection methods can generally be categorized into two main approaches: filter-based methods and wrapper-based methods [2], [3]. In filter methods, the quality assessment of a feature subset is independent of the learning algorithm's performance, particularly the classifier's accuracy. In contrast, wrapper methods evaluate

* Corresponding Author: dalvand.ehsan@gmail.com
Department of Electrical and Computer Engineering, Shahid Beheshti University, Tehran, Iran

http://dx.doi.org/10.47176/TMI.2021.137

feature subsets based on the classification accuracy achieved with test samples [4]. However, results obtained from wrapper methods may not be applicable to other learning algorithms. The primary advantage of wrapper methods is their superior classification accuracy, often surpassing filter-based methods. Nevertheless, their major drawback is the high computational cost [5], as evaluating each feature subset requires training a classifier on the dataset. This issue is particularly problematic for computationally expensive algorithms such as neural networks and support vector machines (SVMs). Due to their high computational costs, wrapper methods are impractical for large-scale datasets.

Traditional filter methods typically eliminate irrelevant features by ranking them based on their relevance to the target class and discarding low-ranking features [2], [6]. However, filter-based ranking methods compare features individually, leading to suboptimal results when the dataset contains many redundant features, thereby reducing classification accuracy [4].

Rough set theory, which is utilized in this study, was introduced by Pawlak in 1982 [7]. This theory approximates a given set using multiple base sets and does not require external parameters for data analysis and inference. It provides tools for evaluating feature subsets and has been successfully employed in search-based feature selection methods to assess the fitness of selected subsets [8,9].

The Intelligent Water Drops (IWD) algorithm is a heuristic search method proposed by Shah-Hosseini [10]. It belongs to the family of swarm intelligence algorithms and is inspired by the natural flow of rivers and water droplets. Observations show that natural rivers tend to find optimal or near-optimal paths among all possible routes. These optimal paths emerge through interactions between water droplets and the riverbed. In recent years, the IWD algorithm has demonstrated significant success in optimization problems [11-13] and machine learning applications [14].

In this study, the Intelligent Water Drops Algorithm is combined with Rough Set Theory for feature selection. The proposed model, named IWDAR, aims to improve the final solution quality while reducing the computational complexity of existing methods. The first section of this paper provides the fundamental concepts of rough set theory for feature selection. The subsequent section presents the new algorithm, followed by experimental results and analysis.

2. ROUGH SET THEORY

In an information system, knowledge and data are stored as a collection of phenomena, instances, and observations. These systems are known as information systems. The attributes collected within an information system are categorized into two groups:

- **Conditional attributes**, which represent experimental results and measurements.
- **Decision attributes**, which reflect expert decisions or event outcomes.

Definition 1: Decision Table

A decision table, denoted as $DT = (U, A, V, f)$, consists of the following components:

- $U = \{x_1, \dots, x_n\}$ is a non-empty set of objects.
- $A = C \cup D$ is a set of attributes, where CCC represents the set of conditional attributes, and D represents the set of decision attributes, such that C∩D=∅.
- $V = U_{a \in A} V_a$, where V_a is the set of possible values for attribute a.
- Each attribute a∈Aa \in Aa∈A defines an information function $f_a : U \rightarrow V_a$.

Definition 2: Indiscernibility Relation

For any subset of attributes R⊆A, an **indiscernibility relation** (denoted as IND(R)) is defined as follows:

$$\mathrm{IND(R)} = \{x, y \in U \times U : a \in R, f(x,a) = f(y,a)\} \quad (1)$$

For every $x \in U$, the **equivalence class** of x under the indiscernibility relation IND(R) is denoted as $[x]_R$. The indiscernibility relation induces a partition on the reference set U, which is represented as U/IND(R) or, in abbreviated form, U/R. The set U/R consists of all equivalence classes generated by the indiscernibility relation.

Definition 3: Lower and Upper Approximations

The lower approximation of a subset X, denoted as $\underline{R}(X)$, is the set of all members of the reference set U that can be definitely classified as members of X with respect to R.

The upper approximation of a subset X, denoted as $\overline{R}(X)$, is the set of all members of U that can be possibly classified as members of X with respect to R.

Definition 4: Positive, Negative, and Boundary Regions

The boundary region $BND_R(X)$ consists of all members that belong to the upper approximation but do not belong to the lower approximation. A set is referred to as rough if its boundary region is non-empty. Rough set theory defines upper and lower approximations to identify the positive region.

In a decision system, the positive region $POS_R(X)$ of the reference set U with respect to R is the set of all members that can be definitely classified using the attributes in R. The negative region of the reference set is the set of all members that cannot be definitively classified using the attributes in R. In other words:

$$\begin{aligned} POS_R(X) &= \underline{R}(X) \\ BND_R(X) &= \overline{R}(X) - \underline{R}(X) \\ NEG_R(X) &= U - \overline{R}(X) \end{aligned} \quad (2)$$

Definition 5: Degree of Dependency

The degree of dependency is defined by the following relation (Equation 3):

$$\gamma_R(D) = \frac{|POS_R(D)|}{|U|} \quad (3)$$

The degree of dependency represents the proportion of members in the reference set U that can be definitely classified into the decision classes U/D. It holds significant importance in feature selection methods that utilize rough set theory, as it can be used as a criterion to evaluate the quality of a subset of features.

Definition 6: Reduction and Core

A reduction is a subset of essential attributes that can fully determine all elements in the dataset. A subset $R \subseteq C$ is called a reduction of C if its dependency degree with respect to D is equal to the dependency degree of CCC with respect to D, and this condition does not hold for any of its proper subsets, i.e., $\gamma_C(D) = \gamma_R(D)$In a decision system, there may exist multiple reductions. The core is defined as the intersection of all reductions. In other words, the core is the set of all attributes that cannot be removed.

Definition 7: Discernibility Matrix

If n is the number of elements in the set U, the discernibility matrix T can be defined as a symmetric $n \times n$ matrix. Each entry C_{ij} in this matrix is determined according to Equation (4).

$$\mathrm{C_{ij}} = \{\mathrm{a} \in \mathrm{C} | \mathrm{a(i)} \neq \mathrm{a(j)}\} \quad , \mathrm{i,j} = 1,2,\dots,\mathrm{n} \quad (4)$$

Any feature that appears individually in an entry of the discernibility matrix is considered an indispensable feature. Therefore, the core C is the set of all such individual features in the discernibility matrix. This matrix can also be used to identify all reductions of C.

In any information system, certain features play a crucial role in data analysis and cannot be removed. The goal of feature selection methods is to retain these essential features while eliminating less significant ones. In other words, in feature selection problems, the objective is to find a subset of features whose dependency degree is equal to that of the full feature set.

An ideal feature selection algorithm seeks to identify all such subsets, a task that is computationally expensive. Therefore, an efficient search algorithm is required to determine an optimal feature subset that maximizes dependency degree while minimizing the subset size.

Feature selection methods using rough sets are generally divided into two categories in the search for reductions of a set of features. The first category includes methods that utilize dependency degree, while the second category comprises methods that employ the discernibility matrix. The first category of methods [15-20] often fails to find an optimal subset due to the absence of an optimal heuristic in this domain. On the other hand, employing the discernibility matrix for high-dimensional datasets entails a high computational cost [21,22]. Consequently, researchers have turned to random search and metaheuristic methods. In this category, algorithms such as Genetic Algorithms [23-25], Ant Colony Optimization [9], [26-28], Particle Swarm Optimization [29-31], and the Intelligent Water Drops (IWD) algorithm [32] have been utilized to obtain optimal solutions. Additionally, in recent years, efforts have been made to address this problem using the Artificial Fish Swarm Algorithm [33-34].

3. INTELLIGENT WATER DROPS FOR FEATURE SELECTION

The proposed IWDAR algorithm is based on the standard IWD algorithm but introduces new features, which are discussed in this section. The algorithm functions such that in each iteration, all water drops generate a solution, and then the soil on all paths is updated based on the best solution constructed. This iteration continues until the stopping criterion is met, which in this case is defined as a maximum number of iterations. The best solution in each iteration is the one containing the fewest number of features.

3.1. Problem Representation

The IWD algorithm requires representing the problem as a graph (N,E), where the nodes correspond to the problem's features, and moving from node i to node j signifies selecting feature j and adding it to the solution set constructed by IWD. Based on this representation, the search for an optimal feature subset is carried out by traversing the graph until the stopping criterion is satisfied.

In Figure 1, which illustrates this representation, the IWD starts at node a, then selects nodes b and c, and finally reaches node d, at which point the stopping criterion is met. The IWD algorithm then terminates its traversal and returns the subset {a,b,c,d} as the constructed solution.

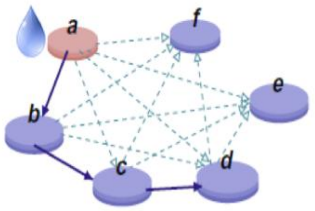

Fig. 1. Representation of the Feature Selection Problem

3.2. Generating a Solution

The process of generating a solution consists of the following steps:

1. **Selecting the Starting Node:** The initial node is chosen.

2. **Choosing the Next Node:** The next node to which the water drop moves is selected.
3. **Updating Parameters:** The velocity of the water drop, the amount of soil carried by it, and the remaining soil on the traversed path are updated.
4. **Checking for a Reduction:** If the set of nodes visited by the water drop (i.e., the selected feature subset) constitutes a reduction, a valid solution is obtained.

3.2.1. Selecting the Starting Node

In most problems, the starting node is selected randomly, and each IWD chooses an initial node randomly. However, in the feature selection problem based on rough set theory, a concept known as the **core** exists, which represents the set of indispensable features. Consequently, each IWD must include the core features in its final solution. Instead of a completely random start, it is assumed that all IWDs begin by selecting the nodes from the core set. Thus, each IWD initially selects a node randomly from the set *C−CORE(C)*.

3.2.2. Selecting the Next Node

An intelligent water drop prefers a path with less soil compared to other available paths. This property is achieved by assigning probabilities to each unvisited path that does not violate the problem's constraints. Suppose an IWD is at node i, then the probability of selecting node j is determined by Equation (5).

$$P_i^{IWD}(j) = \frac{f(\,soil(i,j)\,)}{\sum_{k \notin vc(IWD)} f(\,soil(i,k)\,)} \tag{5}$$

In this relation, *f(soil(i,j))* is the inverse of the amount of soil present between node i and node j.

$$f(soil(i,j)) = \frac{1}{\varepsilon_s + g(\,soil(i,j)\,)} \tag{6}$$

The constant parameter ε_s is a small fixed number introduced to avoid division by zero, typically set to 0.01. $g(soil(i,j))$ is used to shift the amount of soil present between node i and node j toward positive values and is obtained from Equation (7).

$$g(soil(i,j)) = \begin{cases} soil(i,j) & \min_{l \notin vc(IWD)}(soil(i,l)) \geq 0 \\ soil(i,j) - \min_{l \notin vc(IWD)}(soil(i,l)) & Otherwise \end{cases} \tag{7}$$

Where the function min(.) returns the minimum of its argument, and $vc(IWD)$ is the set of nodes that should not be selected, which includes nodes that have already been selected and nodes that violate the problem's constraints.

3.2.3. Velocity Update

Consider an IWD located at node i that intends to move to node j. According to Equation (8), the amount of soil present on the edge (i,j), denoted by soil(i,j), is used to update the velocity of the drop.

$$vel^{IWD}(t+1) = vel^{IWD}(t) + \begin{cases} \frac{a_v}{b_v + c_v . soil(i,j)^{\beta}} & soil(i,j)^{\beta} \neq -\frac{b_v}{c_v} \\ 0 & otherwise \end{cases} \tag{8}$$

In the above relation, $vel^{IWD}(t+1)$ represents the new velocity of the drop after reaching node j, and a_v،b_v , c_v are constant velocity parameters, which, based on initial experiments, have been set to 1000, 0.01, and 1, respectively, with the parameter β assigned a value of 2.

3.2.4. Heuristic Function

In the proposed algorithm, the concept of dependency from rough set theory is used to compute the heuristic value. Suppose an Intelligent Water Drop (IWD) is at node i and selects node j as the next node. Additionally, let V be the set of nodes that the IWD has selected so far, including i.

In this case, the undesirability degree of the path between nodes i and j, denoted as $HUD(i,j)$, is computed using Equation (9).

$$\mathrm{HUD(i,j)} = 1 - \gamma_{(\mathrm{V+\{j\}})}(\mathrm{D}) \tag{9}$$

Where $\gamma_{(V+\{j\})}(D)$ represents the degree of dependency between $V+\{j\}$ and D. The time required for the IWD to move from node i to node j is obtained from Equation (10), where ε_v is a positive constant value, typically set to 0.001.

$$\mathrm{time(i,j;vel^{IWD})} = \frac{1 - \gamma_{(\mathrm{V+\{j\}})}(\mathrm{D})}{\max(\varepsilon_{\mathrm{v}}, \mathrm{vel(t+1)^{IWD}})} \tag{10}$$

3.2.5. Local Soil Update

After selecting the next node and adding it to the set of selected nodes, the amount of soil on the path, as well as the soil carried by the Intelligent Water Drop (IWD), are updated.

The amount of soil removed by the IWD from the edge is computed using Equation (11).

$$\Delta soil(i,j) = \begin{cases} \dfrac{a_s}{b_s + c_s.time(i,j)^{\omega}} & time(i,j)^{\omega} \neq -\dfrac{b_s}{c_s} \\ 0 \end{cases} \tag{11}$$

a_v،b_v , c_v are constant velocity parameters, set to 1000, 0.01, and 1, respectively. The parameter ω is assigned a value of 2, which has the advantage of avoiding division by zero and simplifying the relation. After the IWD moves from node i to node j, the soil on the edge (i,j) must be updated, which is performed using Equation (12).

$$\mathrm{soil(i,j)} = \rho_0.\mathrm{soil(i,j)} - \rho_{\mathrm{n}}.\Delta\mathrm{soil(i,j)} \tag{12}$$

Where in this relation, ρ_0 , ρ_n are local soil update constants and are positive values between 0 and 1. In this paper, the soil update on the path, after a new node is selected, is considered such that $\rho_0 = 1 - \rho_n$. Thus, the relation takes the form of Equation (13).

$$\mathrm{soil(i,j)} = (1 - \rho_{\mathrm{n}}).\mathrm{soil(i,j)} - \rho_{\mathrm{n}}.\Delta\mathrm{soil(i,j)} \tag{13}$$

Additionally, during the transition between two nodes, the amount of soil carried by the IWD must also be added, which is performed using Equation (14).

$$\mathrm{soil^{IWD}} = \mathrm{soil^{IWD}} + \Delta\mathrm{soil(i,j)} \tag{14}$$

3.2.6. Positive Region Computation

After adding a feature to the selected feature set, the next step is to compute the dependency degree of the new feature set and compare it with the dependency degree of the entire feature set concerning the decision set. This requires calculating the number of elements in the positive region $|POS(B)|$, where B is the subset of selected features.

This computation must be performed each time an Intelligent Water Drop (IWD) selects a node, making it one of the computationally expensive steps. The primary challenge in computing the positive region lies in its time complexity.

The basic algorithm for this computation has a time complexity of $O(n^2m)$, where n is the number of samples and mmm is the number of features in the dataset. In [35], an improved algorithm with a complexity of $O(nmlogn)$ was proposed, which remains widely used in methods requiring positive region computation. This algorithm requires $nmlogn + nm$ computations, and for constructing a solution of length L, the total number of required computations is determined by Equation (15).

$$\mathrm{T} = (\mathrm{nlogn} + \mathrm{n}) + 2 \times (\mathrm{nlogn} + \mathrm{n}) + \cdots + \mathrm{L} \times (\mathrm{nlogn} + \mathrm{n}) = \frac{\mathrm{L} \times (L-1)}{2} \times (\mathrm{nlogn} + \mathrm{n}) \quad (15)$$

Despite the improved time complexity, the computational burden remains a challenge, making the final algorithm require significant time to compute the minimal reduction. To address this issue, this paper proposes a new approach for computing the positive region based on the stepwise and incremental nature of the IWDAR algorithm. The key idea is to reuse previously computed results at each stage, thereby avoiding redundant calculations and achieving a more acceptable time complexity.

According to the proposed approach in IWDAR, the number of required computations for constructing a solution of length L is reduced to 2L×(nlogn+n).

The proposed algorithm first sorts the dataset based on the feature set B and then computes the positive region.

Consider an Intelligent Water Drop (IWD) currently at node ccc, having selected the features a, b, and ccc so far. In the next step, it selects feature d to add to the solution set. The task now is to compute the positive region for the subset B={a,b,c,d}.

The dataset is sorted incrementally, first by feature d, then by feature ccc, and so on until the first feature, a. However, in the previous step, when the IWD was at node b and selected feature ccc, the sorting had already been performed based on ccc, b, and a. Therefore, the results of the previous step can be reused, and sorting needs to be performed only for the newly added feature.

One way to implement this incremental sorting is by storing the indices from the sorting results of the previous step. The index array, denoted as X, has a length of n (where n is the number of samples). Each element X(i) represents the row position of sample i in the previously sorted dataset.

In the proposed algorithm, whose pseudocode is presented in Algorithm 1, an array D is used to store results related to discernibility classes, further optimizing computation.

In Algorithm 1, the first step is to check whether the results from the previous step can be reused. This decision is based on the presence or absence of the input array D.

- If this is the first sorting operation, the dataset is sorted based on the input feature set using Algorithm 2.
- After the initial sorting, the main loop of Algorithm 1 iterates through all samples to determine equivalence classes.

The algorithm identifies which samples have identical values in the input feature vector. This is done by:

1. Iterating top-down through the dataset matrix.
2. Comparing feature values of each sample with the previous samples.

Since the dataset has already been sorted, all samples with identical feature values appear consecutively. The process continues until encountering a sample with a different feature value, at which point a new equivalence class is formed, and the same procedure is applied.

- All samples within an equivalence class are assigned the same $\boldsymbol{D'}$ value, which represents their class.
- The D' vector can then be used in the next computation step.

3.2.7. Computing the Positive Region

Once all samples have been processed in the first loop of Algorithm 1, the next step is to compute the number of elements in the positive region. This is handled by the second loop in the same algorithm:

- If all samples in an equivalence class have the same decision attribute value, they are added to the positive region [35].

In the proposed sorting step, instead of sorting the entire dataset based on all input features, we perform two sorts: one based on the D vector and the other based on the last added feature (feature d in the example provided).

This approach reduces the number of required sorting operations from *mnlogn* to *2nlogn*.

After this optimized sorting, to compute the equivalence classes for the new subset, we only need to compare the sample values based on:

1. The last added feature
2. The equivalence class from the previous step.

As a result, the number of computations is independent of the subset's length and requires only 2n operations.

3.3. Best Solution Selection and Global Soil Update

After all Intelligent Water Drops (IWDs) have constructed their solutions and reached a reduction of the feature set, the best solution from the iteration T^{IB} must be computed. This requires a fitness function.

Since the output of all IWDs is a reduction, the best solution is the reduction with the fewest elements. Based on this criterion, T^{IB} is computed using Equation (16).

$$\mathrm{T^{IB}} = \arg\min_{\forall \mathrm{T^{IWD}}}(|\mathrm{T^{IWD}}|) \tag{16}$$

After identifying the best solution in each iteration, the soil on the paths of this solution must be updated. In the IWD algorithm, Equation (17) is used for the global soil update.

$$\mathrm{soil(i,j)} = \rho_s.\mathrm{soil(i,j)} - \frac{\rho_{\mathrm{IWD}}}{\mathrm{N_C}-1}.\mathrm{soil_{IB}^{IWD}} \qquad \forall(\mathrm{i,j}) \in \mathrm{T^{IB}} \tag{17}$$

In this context, $soil_{IB}^{IWD}$ represents the amount of soil associated with the Intelligent Water Drop (IWD) that created the best solution in the iteration.

- N_C is the number of nodes in the best solution.
- ρ_s , ρ_{IWD} are constant parameters used for global soil update.

In the AWDAR algorithm, after performing initial experiments, to ensure the best solutions are not lost due to the positive or negative changes in the soil on the path (depending on the coefficients in the equation), these coefficients are set equal to 1. As a result, Equation (17) is simplified to Equation (18).

$$\mathrm{soil(i,j)} = \mathrm{soil(i,j)} - \frac{\mathrm{soil_{IB}^{IWD}}}{\mathrm{N_C}-1} \qquad \forall(\mathrm{i,j}) \in \mathrm{T^{IB}} \tag{18}$$

Another important consideration is which edges to select for the update. In the original algorithm [11], the update is performed for the edges that the Intelligent Water Drops (IWDs) have traversed, based on the order in which the nodes are added in the best solution. However, in the feature selection problem, the solution created by each IWD is a subset of features, and the order in which features are added is not important.

Thus, in the soil update process, all permutations of T^{IB} are considered. In other words, the update is applied to all edges that connect the features present in the solution set (as shown in Algorithm 3).

At the end of each iteration, it is necessary to check whether a better solution has been found compared to previous iterations. If a better solution is found, the current best solution in the algorithm (denoted as T^{IB}) must be updated. This update is carried out using Equation (19).

$$\mathrm{T^{TB}} = \begin{cases} \mathrm{T^{TB}} & \mathrm{q(T^{TB})} \geq \mathrm{q(T^{IB})} \\ \mathrm{T^{IB}} & Otherwise \end{cases} \tag{19}$$

4. EXPERIMENTS AND RESULTS

In this section, the experimental results of the conducted tests are presented and analyzed. The IWDAR algorithm was tested on a personal computer with a 2.27 GHz Core i3 processor and 4 GB of RAM. Additionally, Matlab R2009a was used for implementing the algorithms.

The IWDAR algorithm was tested on 17 datasets, whose specifications are provided in Table 1, and the results were compared with other methods. Among these datasets, M-of-N, Exactly, and Exactly2 are based on the work in [36], while the remaining datasets are from the University of California, Irvine dataset repository.

Table 1. Specifications of the Datasets Used

Dataset Name	Number of Features	Number of Samples	Number of Core Features
Monk3	6	432	3
Breast-cancer	9	699	1
M-of-N	13	1000	6
Exactly	13	1000	6
Exactly2	13	1000	10
Heart	13	294	3
Zoo	16	101	2
Vote	16	300	7
Credit	20	1000	1
Mushroom	22	8124	-
Led	24	2000	2
Letter	25	26	-
Derm	34	366	-
Chess-king	36	3196	27
WQ	38	521	-
Lung	56	32	-
Audiology	69	200	3

4.1. Comparison of IWDAR Results with the IWDRSAR Algorithm

As previously mentioned, in [32], the IWD algorithm was used to solve the feature selection problem using fuzzy sets. The name of this algorithm is IWDRSAR. In Table 2, the values of the constant parameters used in both algorithms are shown.

Table 2. Starting Parameters of the IWDAR Algorithm

Values in IWDAR	Values in IWDRSAR	Parameters	Description
Number of features minus core	Number of features minus core	N_{IWD}	**Static Parameters**
1000, 0.01, 1	1000, 0.01, 1	a_v, b_v, c_v	
1000, 0.01, 1	1000, 0.01, 1	a_s, b_s, c_s	
100	100	initSoil	
250	250	maxIter	
0.6	0.9	ρ_n	
1	0.1, 0.9	ρ_s, ρ_{IWD}	
Empty	Empty	$vc(IWD)$	**Dynamic Parameters**
4	4	initVel	
0	0	$soil(IWD)$	

4.2. Solution Quality

Since random search methods may yield different solutions in each iteration, as suggested in [9], both algorithms were executed 20 times on the datasets, and the results are presented in Table 3.

As can be seen, the IWDAR algorithm performs significantly better than IWDRSAR across all examined datasets. The better output refers to the number of selected features in the reduced set and the number of times the minimum reduction was achieved during the 20 runs.

A notable observation in this table is the Derm dataset, where the IWDRSAR algorithm failed to converge correctly. Not only did it achieve the minimum length reduction (6) only 2 times, but it also returned reductions of 7, 8, 9, and 10 as the best solutions 3, 5, 5, and 5 times, respectively. This diversity in output indicates a lack of proper convergence, which was addressed in the IWDAR algorithm through parameter optimization and the global soil update rule, which is specific to the feature selection problem.

Table 3. Comparison of Results from IWDRSAR and IWDAR Algorithms in Terms of Output Quality

IWDAR	IWDRSAR	Minimal Reduction	Number of Features	Dataset Name
3	$(3)^{(18)}$، $(4)^{(2)}$	3	6	**Monk3**
4	**4**	4	9	**Breast-cancer**
6	$(6)^{(18)}$، $(7)^{(2)}$	6	13	**M-of-N**
6	$(6)^{(16)}$، $(7)^{(4)}$	6	13	**Exactly**
10	**10**	10	13	**Exactly2**
6	**6**	6	13	**Heart**
5	**5**	5	16	**Zoo**
8	$(8)^{(9)}$، $(9)^{(11)}$	8	16	**Vote**
8	$(10)^{(12)}$، $(11)^{(8)}$	8	20	**Credit**
4	$(4)^{(16)}$، $(5)^{(4)}$	4	22	**Mushroom**
5	$(5)^{8)}$، $(6)^{(8)}$، $(7)^{(4)}$	5	24	**Led**
8	$(8)^{(16)}$، $(9)^{(4)}$	8	25	**Letter**
6	$(6)^{(2)}$، $(7)^{(3)}$ ،$(8)^{(5)}$ ،$(9)^{(5)}$ ،$(10)^{(5)}$	6	34	**Derm**
$(29)^{(18)}$ ،$(30)^{(2)}$	$(30)^{(6)}$، $(31)^{(10)}$، $(32)^{(4)}$	29	36	**Chess-king**
$(12)^{(5)}$، $(13)^{(15)}$	$(13)^{(3)}$، $(14)^{(17)}$	12	38	**WQ**
$(4)^{(17)}$، $(5)^{(3)}$	$(4)^{(6)}$، $(5)^{(12)}$، $(6)^{(2)}$	4	56	**Lung**
$(12)^{(3)}$، $(13)^{(16)}$ $(14)^{(1)}$	$(20)^{(4)}$، $(21)^{(6)}$ ،$(22)^{(8)}$ ،$(23)^{(2)}$	-	69	**Audiology**

4.3. Comparison of IWDAR with Other Methods

In this section, the results of the IWDAR algorithm are compared with five other algorithms: AntRSAR [9], GenRSAR [9], TSAR [31], ACOAR [26], and RSFSACO [27], using the datasets mentioned earlier.

As observed, IWDAR outperforms three of the other methods in all datasets. In Table 6, the results of the proposed method are compared with two methods, AntRSAR and ACOAR, both of which utilize the Ant Colony Optimization (ACO) algorithm for the search process.

Table 4. Comparison of IWDAR Results with Results of GenRSAR, SimRSAR, and TSAR Methods

IWDAR	TSAR	SimRSAR	GenRSAR	Number of Features	Dataset
6	**6**	**6**	$(6)^{(8)}$، $(7)^{(12)}$	13	**M-of-N**
6	**6**	**6**	$(6)^{(10)}$، $(7)^{(10)}$	13	**Exactly**
10	**10**	**10**	$(10)^{(9)}$، $(11)^{(11)}$	13	**Exactly2**
6	**6**	$(6)^{(19)}$، $(7)^{(1)}$	$(6)^{(18)}$، $(7)^{(2)}$	13	**Heart**
8	**8**	$(8)^{(15)}$، $(9)^{(5)}$	$(8)^{(2)}$، $(9)^{(18)}$	16	**Vote**
8	$(8)^{(13)}$، $(9)^{(5)}$ ،$(10)^{(2)}$	$(8)^{(18)}$، $(9)^{(1)}$ ،$(10)^{(1)}$	$(10)^{(6)}$، $(11)^{(14)}$	20	**Credit**
4	$(4)^{(17)}$، $(5)^{(3)}$	**4**	$(6)^{(5)}$ ،$(7)^{(14)}$ $(5)^{(1)}$،	22	**Mushroom**
5	**5**	**5**	$(7)^{(3)}$ ،$(8)^{(16)}$ $(6)^{(1)}$،	24	**Led**
8	$(8)^{(17)}$، $(9)^{(3)}$	**8**	$(8)^{(8)}$، $(9)^{(12)}$	25	**Letter**
6	$(6)^{(14)}$، $(7)^{(6)}$	$(6)^{(12)}$، $(7)^{(8)}$	$(10)^{(6)}$، $(11)^{(14)}$	34	**Derm**
(12)$^{(5)}$، (13)$^{(15)}$	$(12)^{(1)}$ ،$(13)^{(13)}$،$(14)^{(6)}$	$(13)^{(16)}$،$(14)^{(4)}$	16	38	**WQ**
(4)$^{(17)}$، (5)$^{(3)}$	$(4)^{(6)}$، $(5)^{(13)}$ ،$(6)^{(1)}$	$(4)^{(7)}$، $(5)^{(12)}$ ،$(6)^{(1)}$	$(6)^{(8)}$، $(7)^{(12)}$	56	**Lung**

Table 5. Comparison of IWDAR Results with Results of AntRSAR and ACOAR Methods

IWDAR	ACOAR	AntRSAR	Number of Features	Dataset
6	**6**	**6**	13	**M-of-N**
6	**6**	**6**	13	**Exactly**
10	**10**	**10**	13	**Exactly2**
6	**6**	$(6)^{(18)}$، $(7)^{(2)}$	13	**Heart**
8	**8**	**8**	16	**Vote**
8	$(8)^{(16)}$، $(9)^{(4)}$	$(8)^{(12)}$، $(9)^{(4)}$ ،$(10)^{(4)}$	20	**Credit**
4	**4**	$(4)^{(13)}$، $(5)^{(7)}$	22	**Mushroom**
5	**5**	$(5)^{(12)}$، $(6)^{(4)}$ ،$(7)^{(4)}$	24	**Led**
8	**8**	**8**	25	**Letter**
6	**6**	$(6)^{(17)}$، $(7)^{(3)}$	34	**Derm**
(12)$^{(5)}$، (13)$^{(15)}$	$(12)^{(4)}$ ،$(13)^{(12)}$،$(14)^{(4)}$	$(12)^{(2)}$ ،$(13)^{(7)}$،$(14)^{(11)}$	38	**WQ**
$(4)^{(17)}$، $(5)^{(3)}$	**4**	**4**	56	**Lung**

IWDAR has generally shown better performance compared to the other two methods, and only in the Lung dataset has it failed to achieve the best possible output. Another method that uses the Ant Colony Optimization algorithm is RSFSACO. In this study, we have compared the results obtained from IWDAR with those from RSFSACO and also the AntRSAR method across seven datasets.

Table 6. Comparison of IWDAR Results with AntRSAR and RSFSACO Methods

IWDAR	RSFSACO	AntRSAR	Number of Features	Dataset
(12)$^{(3)}$، (13)$^{(16)}$ (14)$^{(1)}$	$(13)^{(4)}$، $(14)^{(16)}$	$(20)^{(2)}$، $(21)^{(18)}$	70	Audiology
4	**4**	**4**	13	Breast
$(29)^{(18)}$ ،$(30)^{(2)}$	**29**	$(29)^{(4)}$ ،$(30)^{(11)}$،$(31)^{(5)}$	13	Chess-king
4	**4**	**4**	13	Monk3
4	$(4)^{(13)}$، $(5)^{(7)}$	$(4)^{(13)}$، $(5)^{(7)}$	22	Mushroom
8	9	$(10)^{(5)}$ ،$(11)^{(7}$،$(12)^{(8)}$	16	Vote
5	**5**	$(5)^{(18)}$، $(6)^{(2)}$	16	Zoo

The overall performance of IWDAR is once again superior to the other two methods, with the only exception being the Chess-king dataset, where it did not achieve the best result.

5. CONCLUSION

In this study, an optimized approach for addressing the feature selection problem is proposed by integrating the Intelligent Water Drops (IWD) algorithm with rough set theory. The developed algorithm, referred to as IWDAR, possesses the following key characteristics:

- It leverages the principles of rough set theory alongside the IWD algorithm.
- It incorporates results from previous iterations to minimize computational costs.
- It initiates its search from the core set of features, thereby simplifying both the problem graph and the search space.
- By optimizing the rules and parameters of the IWD algorithm for feature selection, the proposed approach enhances computational efficiency.

The findings of this study demonstrate that the IWD algorithm outperforms other tested metaheuristic search methods in feature selection tasks, making it a reliable tool for analyzing high-dimensional datasets. Moreover, the results suggest that incorporating elitism and emphasizing the best solutions obtained in each iteration can improve the efficiency of metaheuristic-based feature selection methods. This, in turn, reduces the likelihood of algorithm stagnation and non-convergence. Additionally, the study highlights the potential of rough set theory as an effective evaluation criterion, not only for feature selection but also for problem graph simplification. The results further indicate that computational cost, a major limitation of rough set-based discovery, can be mitigated to some extent through the proposed approach. While the IWDAR algorithm has demonstrated competitive performance compared to other existing methods on the tested datasets, its effectiveness in real-world applications and decision-making accuracy require further assessment. A primary limitation of this method, as with all approaches utilizing rough set theory, is its reliance on discrete feature values. To address this issue, future research could explore extensions of rough set theory, such as fuzzy rough sets. Despite the improvements in computational efficiency, the algorithm still encounters significant challenges when applied to datasets with thousands of features, primarily due to the high computational cost associated with the discovery function. Consequently, further refinement of this aspect is necessary to enhance the algorithm's scalability and practical applicability.

Transparency Statement

The data supporting this study are available upon reasonable request to the corresponding author, subject to ethical and confidentiality considerations.

Acknowledgments

We would like to express our gratitude to all individuals who contributed to this project.

Declaration of Interest

The authors declare that they have no competing interests.

Funding

This research received no specific grant from any funding agency, commercial, or not-for-profit sectors.

REFERENCES

[1] Kabir, M. M., Shahjahan, M., & Murase, K. (2011). A new local search based hybrid genetic algorithm for feature selection. Neurocomputing, 74(17), 2914-2928. https://doi.org/10.1016/j.neucom.2011.03.034

[2] Das, S. (2001). Filters, wrappers and a boosting-based hybrid for feature selection. In Proceedings of the Eighteenth International Conference on Machine Learning (pp. 74-81).

[3] Fleuret, F. (2004). Fast binary feature selection with conditional mutual information. Journal of Machine

Learning Research, 5(Nov), 1531-1555.

[4] Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. Artificial Intelligence, 97(1-2), 273-324. https://doi.org/10.1016/S0004-3702(97)00043-X

[5] Kudo, M., & Sklansky, J. (2000). Comparison of algorithms that select features for pattern classifiers. Pattern Recognition, 33(1), 25-41. https://doi.org/10.1016/S0031-3203(99)00041-2

[6] Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. Journal of Machine Learning Research, 3(Mar), 1157-1182.

[7] Pawlak, Z. (1982). Rough sets. International Journal of Computer & Information Sciences, 11(5), 341-356. https://doi.org/10.1007/BF01001956

[8] Suguna, N., & Thanushkodi, K. (2010). A novel rough set reduct algorithm for medical domain based on bee colony optimization.

[9] Jensen, R., & Shen, Q. (2003). Finding rough set reducts with ant colony optimization. In Proceedings of the 2003 UK Workshop on Computational Intelligence (pp. 15-22).

[10] Shah-Hosseini, H. (2007). Problem solving by intelligent water drops. In 2007 IEEE Congress on Evolutionary Computation (pp. 3226-3231). https://doi.org/10.1109/CEC.2007.4424885

[11] Shah-Hosseini, H. (2008). Intelligent water drops algorithm: A new optimization method for solving the multiple knapsack problem. International Journal of Intelligent Computing and Cybernetics, 1(2), 193-212. https://doi.org/10.1108/17563780810874717

[12] Shah-Hosseini, H. (2009). The intelligent water drops algorithm: A nature-inspired swarm-based optimization algorithm. International Journal of Bio-Inspired Computation, 1(1/2), 71-79. https://doi.org/10.1504/IJBIC.2009.022775

[13] Niu, S. H., Ong, S. K., & Nee, A. Y. C. (2012). An improved intelligent water drops algorithm for achieving optimal job-shop scheduling solutions. International Journal of Production Research, 50(15), 4192-4205. https://doi.org/10.1080/00207543.2011.600346

[14] Shah-Hosseini, H. (2012). Intelligent water drops algorithm for automatic multilevel thresholding of grey-level images using a modified Otsu's criterion. International Journal of Modelling, Identification and Control, 15(4), 241-251. https://doi.org/10.1504/IJMIC.2012.046402

[15] Modrzejewski, M. (1993). Feature selection using rough sets theory. In Proceedings of the European Conference on Machine Learning (pp. 213-226). https://doi.org/10.1007/3-540-56602-3_138

[16] Zhong, N., Dong, J., & Ohsuga, S. (2001). Using rough sets with heuristics for feature selection. Journal of Intelligent Information Systems, 16(3), 199-214.

[17] Chouchoulas, A., & Shen, Q. (2001). Rough set-aided keyword reduction for text categorization. Applied Artificial Intelligence, 15(9), 843-873. https://doi.org/10.1080/088395101753210773

[18] Jensen, R., & Shen, Q. (2004). Fuzzy-rough attribute reduction with application to web categorization. Fuzzy Sets and Systems, 141(3), 469-485. https://doi.org/10.1016/S0165-0114(03)00021-6

[19] Mac Parthalain, N., Shen, Q., & Jensen, R. (2007). Distance measure assisted rough set feature selection. In 2007 IEEE International Fuzzy Systems Conference (pp. 1-6). https://doi.org/10.1109/FUZZY.2007.4295518

[20] Peng, R., Hao, T., & Fang, Y. (2021). Syntax-aware neural machine translation directed by syntactic dependency degree. *Neural Computing and Applications, 33*, 16609-16625. https://doi.org/10.1007/s00521-021-06256-4

[21] Sowkuntla, P., & Prasad, P. (2021). MapReduce-based parallel fuzzy-rough attribute reduction using a discernibility matrix. *Applied Intelligence, 52*, 154-173. https://doi.org/10.1007/s10489-021-02253-1

[22] Jensen, R., & Shen, Q. (2008). Computational intelligence and feature selection: Rough and fuzzy approaches. Wiley. https://doi.org/10.1002/9780470377888

[23] Wróblewski, J. (1995). Finding minimal reducts using genetic algorithms. In Proceedings of the Second Annual Joint Conference on Information Sciences (pp. 186-189).

[24] Zhai, L.-Y., Khoo, L.-P., & Fok, S.-C. (2002). Feature extraction using rough set theory and genetic algorithms An application for the simplification of product quality evaluation. Computers & Industrial Engineering, 43(4), 661-676. https://doi.org/10.1016/S0360-8352(02)00131-6

[25] Hedar, A., Omar, M. A., & Sewisy, A. A. (2015). Rough sets attribute reduction using an accelerated genetic algorithm. In 16th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD) (1-7). https://doi.org/10.1109/SNPD.2015.7176207

[26] Ke, L., Feng, Z., & Ren, Z. (2008). An efficient ant colony optimization approach to attribute reduction in rough set theory. Pattern Recognition Letters, 29(9), 1351-1357. https://doi.org/10.1016/j.patrec.2008.02.006

[27] Chen, Y., Miao, D., & Wang, R. (2010). A rough set approach to feature selection based on ant colony optimization. Pattern Recognition Letters, 31(3), 226-233. https://doi.org/10.1016/j.patrec.2009.10.013

[28] Forsati, R., Moayedikia, A., Jensen, R., Shamsfard, M., & Meybodi, M. R. (2014). Enriched ant colony optimization and its application in feature selection. Neurocomputing, 142, 354-371. https://doi.org/10.1016/j.neucom.2014.03.053

[29] Wang, X., Yang, J., Teng, X., Xia, W., & Jensen, R. (2007). Feature selection based on rough sets and particle swarm optimization. Pattern Recognition Letters, 28(4), 459-471. https://doi.org/10.1016/j.patrec.2006.09.003

[30] Bae, C., Yeh, W. C., Chung, Y. Y., & Liu, S. L. (2010). Feature selection with intelligent dynamic swarm and rough set. Expert Systems with Applications, 37(10), 7026-7032. https://doi.org/10.1016/j.eswa.2010.03.016

[31] Hedar, A. R., Wang, J., & Fukushima, M. (2008). Tabu search for attribute reduction in rough set theory. Soft Computing, 12(9), 909-918. https://doi.org/10.1007/s00500-007-0260-1

[32] Alijla, B. O., Peng, L. C., Khader, A. T., & Al-Betar, M. A. (2013). Intelligent water drops algorithm for rough set feature selection. In Lecture Notes in Computer Science (LNCS, vol. 7803, Part 2) (pp. 356-365). https://doi.org/10.1007/978-3-642-36543-0_37

[33] Su, Y., & Guo, J. (2017). A novel strategy for minimum attribute reduction based on rough set theory and fish swarm algorithm. Computational Intelligence and Neuroscience, 2017, 1-7. https://doi.org/10.1155/2017/6573623

[34] Wang, F., Xu, J., & Li, L. (2014). A novel rough set reduct algorithm to feature selection based on artificial fish swarm algorithm. In Springer, Cham (pp. 24-33). https://doi.org/10.1007/978-3-319-11897-0_4

[35] Hoa, N. S., & Hoa, N. S. (1996). Some efficient algorithms for rough set methods. In Proceedings of IPMU'96 Granada, Spain (pp. 1541-1457).

[36] Ioerger, T. R. (2002). Instance-based filter for feature selection. Journal of Machine Learning Research, 1, 1-23.

APPENDIX

Algorithm 1 - Calculating the Positive Region Based on Previous Results

***Input**: IS information system matrix, B feature subset, D discernibility class array, L last feature*

***Output**: $|POS(B)|$, D' new discernibility class array*

P= $|POS(B)| = 0$

If D=∅ then

Sort IS based on B using algorithm 3

D=an array of ones with length of n, L=B

else

Sort IS based on L , D using algorithm 3

end

k=1, classNum=1

for all objects i in IS do

if IS(i,L)!=IS(k,L) or D(i)!=D(k)

k=i

classNum=classNum+1

end

D'(i)=classNum /set discernibility class i to classNum

end

for all discernibility classes D'(i)

If all objects with same $D'(i)$ *have the same decision value*

then $P = P + number\ of\ objects\ with\ same\ D'(i)$

end

$|POS(B)| = P$

Algorithm 2 - Sorting the Datasets

***Input**: IS, B, D*

***Output**: sorted IS*

If D=∅

Sort IS based on B

else

Index=index of sorting D array

Sort IS(Index) based on B

End

Algorithm 3 - Global Soil Update

```
for  i=1 to j-1 do
   for  k=i+1 to j do
      update soil(i,l)
      soil(l,i)=soil(i,l)
```