JOIV : Int. J. Inform. Visualization, 7(2) - June 2023 477-486

INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION

journal homepage : www.joiv.org/index.php/joiv

Inversed Control Parameter in Whale Optimization Algorithm and Grey Wolf Optimizer for Wrapper-Based Feature Selection: A Comparative Study

Li Yu Yab [a], Noorhaniza Wahid [a,*], Rahayu A Hamid [a]

[a] *Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Parit Raja 86400, Johor, Malaysia*

Corresponding author: [*]nhaniza@uthm.edu.my

***Abstract*—Whale Optimization Algorithm (WOA) and Grey Wolf Optimizer (GWO) are well-perform metaheuristic algorithms used by various researchers in solving feature selection problems. Yet, the slow convergence speed issue in the Whale Optimization Algorithm and Grey Wolf Optimizer could demote the performance of feature selection and classification accuracy. Therefore, to overcome this issue, a modified WOA (mWOA) and modified GWO (mGWO) for wrapper-based feature selection were proposed in this study. The proposed mWOA and mGWO were given a new inversed control parameter expected to enable more search areas for the search agents in the early phase of the algorithms, resulting in a faster convergence speed. This comparative study aims to investigate and compare the effectiveness of the inversed control parameter in the proposed methods against the original algorithms in terms of the number of selected features and the classification accuracy. The proposed methods were implemented in MATLAB where 12 datasets with different dimensionality from the UCI repository were used. kNN was chosen as the classifier to evaluate the classification accuracy of the selected features. Based on the experimental results, mGWO did not show significant improvements in feature reduction and maintained similar accuracy as the original GWO. On the contrary, mWOA outperformed the original WOA regarding the two criteria mentioned, even on high-dimensional datasets. Evaluating the execution time of the proposed methods, utilizing different classifiers, and hybridizing proposed methods with other metaheuristic algorithms to solve feature selection problems would be future works worth exploring.**

***Keywords*— Feature selection; metaheuristics; whale optimization algorithm; grey wolf optimizer; control parameter; high-dimensional dataset.**

Manuscript received 26 Dec. 2022; revised 14 Jan. 2023; accepted 8 Feb. 2023. Date of publication 30 Jun. 2023.

I. INTRODUCTION

Feature selection has been a popular topic for researchers over the past decades. It is known as the process of reducing dimensionality by selecting the most relevant features and discarding less relevant features in a dataset [1]. The objective of such dimensionality reduction is due to the ever-growing dataset dimensionality rate throughout technological advancement. A high-dimensional dataset (HDD) is a matrix consisting of many columns or rows representing a huge size of features or instances [2]. Generally, not every feature could provide significant information for classification, and the less relevant features in a dataset could demote classification accuracy. Especially for HDDs, irrelevant features cause memory constraints and result in the expensive cost of training and computing, which is also known as the "curse of dimensionality" [2], [3]. Therefore, feature selection is often related to two main tasks, that is, to promote maximum classification accuracy and to select a minimum number of selected features to avoid the "curse of dimensionality" [4]–[7]. Besides, feature selection can be interpreted as the pre-processing process for data classification, where the class label of an instance is assigned based on the learning model trained [8]. Since the training data is contributed from the features selected, the ability to select fewer features without lowering the classification accuracy is the criteria to evaluate the effectiveness of a feature selection method.

There are three feature selection methods: filter-based, embedded-based, and wrapper-based [9]. The filter-based method carries out feature selection without using any learning algorithms, and it solely depends on the mutual information provided by the features and their relationship with the class label [10]. After the filter-based feature selection process takes place, any classifiers can be utilized to

evaluate the quality of selected features in a dataset. Therefore, the filter-based method is flexible to implement and fast to execute. Besides, the embedded-based method performs feature selection during the classification process; hence, it can produce high classification accuracy [9]. Unlike filter-based and embedded-based methods, the wrapper-based method uses a certain classifier to obtain the quality of each feature subset. Thus, it has higher time complexity and higher classification accuracy [9]. Therefore, the wrapper-based method feature selection is the focus of this study.

Over the years, metaheuristic optimization algorithms have been utilized with feature selection. In metaheuristic algorithms, exploration and exploitation phases determine the search mechanism to obtain the optimal solution [11]. The exploration phase is where the algorithm randomly looks for potential areas in the search space, whereas the exploitation phase is where the algorithm further scrutinizes a certain search space area found during exploration. These two phases are further discussed in Section 2. Some of the recent metaheuristic optimization algorithms are Manta Ray Foraging Optimization [12], Harris Hawks Optimization [13], Grey Wolf Optimizer (GWO) [14], and Whale Optimization Algorithm (WOA) [15]. Many researchers have employed these metaheuristic algorithms in various domains, such as solving medical problems [16], electromagnetic problems [17], predicting the rate of traffic congestion [18], and handling feature selection problems [10], [19]–[21].

Notably, WOA and GWO have shown similarities in terms of the parameters and characteristics in their algorithm designs, which will be explained in Section 2. These two algorithms are excellent optimizers and have yielded great performance in both filter-based and wrapper-based feature selection methods due to their strengths in exploration and exploitation [10], [19], [20], [22]–[24]. Al-Tashi *et al.* implemented a wrapper-based feature selection with GWO to select the most relevant features to be used in diagnosing coronary artery disease with a two-stage approach [22]. The most relevant features in the Cleveland Heart disease dataset were identified in the first stage, while the fitness function was evaluated by the Support Vector Machine (SVM) classifier in the second stage. Based on the experimental results, the proposed GWO method obtained 89.83%, 93%, and 91% for classification accuracy, sensitivity, and specificity, respectively.

Another research by Hu, Pan, and Chu. [23] implemented a wrapper-based feature selection using a binary variant of GWO (BGWO) and another improved BGWO variant (ABGWO). The binary conversion was achieved by using four V-shaped transfer functions so that the continuous values were mapped to binary values, whereas a new control parameter in ABGWO was utilized to improve the convergence speed of BGWO. The classifier used to obtain classification accuracy in the study is the K-nearest neighbor (kNN). Based on the experimental results, with improved convergence speed, the proposed ABGWO outperformed BGWO in both classification accuracy and execution time for most cases.

Mafarja and Mirjalili [19], [20] introduced two wrapper-based feature selection studies with WOA. Both studies have produced outstanding classification accuracy using the kNN classifier. In the first study, WOA was improved by Tournament selection (WOA-T), roulette wheel selection (WOA-R), as well as the crossover and mutation operators (WOA-CM) [19]. The execution time of WOA-CM was the least as compared to other methods. The average feature selection ratio of WOA-CM was smaller than WOA, and the average classification accuracy of WOA-CM was higher than WOA in most datasets [19]. In the second study, WOA was hybridized with Simulated Annealing (WOA-SA), where WOA handled the global search, whereas SA handled the local search [20]. In addition, WOA-SA with the Tournament selection mechanism (WOA-SAT) was also introduced in the same study. The average feature selection ratio of WOA-SAT was lower than WOA, while the average classification accuracy of WOA-SAT was greater than WOA in most datasets [20].

Nematzadeh *et al.* [10] implemented a filter-based feature selection method with WOA and Mutual Congestion that applied different rates for discarding the feature size starting from 20% to 80% on four medical HDDs. Using the SVM, Naïve Bayes, and Decision Tree classifiers to evaluate the quality of the selected features, the proposed method yielded the highest average classification accuracy when 50% of features from the original HDDs were discarded. The promising results from Nematzadeh *et al.* [10] have inspired the work of another filter-based feature selection method by Yab, Wahid, and Hamid [24] with modified WOA (mWOA). The study adopted a similar approach of using a 50% feature selection size for dimensionality reduction and introduced a new control parameter formula in mWOA to improve the convergence speed of WOA.

According to the literature, the slow convergence speed in WOA is a known issue, as the problem is also found in GWO [25]. GWO is compared to WOA as they have similar characteristics and parameters [23], [26]. In both WOA and GWO, the control parameter, a, determines the trade-off for the exploration and exploitation phases [11], [25], [27]. Since the value of a affects the coefficient vector, $\vec{A}$, it contributes to the distance between the search agents and the prey over iterations. As a result, the global and local search mechanisms that decide when to converge to the optimal solution, are affected by a. Therefore, to reduce the impact of slow convergence speed, the modification of the control parameter's formula is required [11], [28]. Previous studies reported that ABGWO for wrapper-based feature selection [23] and mWOA for filter-based feature selection [24] have both improved the convergence speed. However, the performance of filter-based mWOA has only been tested against filter-based WOA and without feature selection (NO FS).

Therefore, this study presents a comparative study of (i) a proposed mWOA as a wrapper-based feature selection and (ii) a modified GWO (mGWO) by inversing its control parameter formula as in mWOA. This comparative study aims to evaluate the effectiveness of the modified control parameter's formula for both mWOA and mGWO against the original GWO and WOA. The proposed wrapper-based mWOA and mGWO were implemented in MATLAB, and the kNN classifier was used to evaluate the significance of the selected features toward the classification accuracy.

The rest of the paper is organized as follows. Section 2 presents the materials and methods used in this work, which

covers the inspiration and mathematical equations for original WOA and GWO, and introduces the proposed method involving the inversed control parameter for mWOA and mGWO. The experimental results are tabulated and thoroughly discussed in Section 3. Lastly, Section 4 concludes the findings of this comparative study and provides insights into the potential future work.

II. Materials and Method

This section presents the materials used in this study: Whale Optimization Algorithm and Grey Wolf Optimizer. The proposed method and experimental setups used in realizing the comparative study are also included in this section.

A. Whale Optimization Algorithm

Whale Optimization Algorithm (WOA) is a nature-inspired metaheuristic algorithm introduced by Mirjalili and Lewis [15] in 2016 for solving optimization problems. WOA has shown its spectacular performance in various research areas such as engineering, transportation, and medical diagnosis [16], [18], [19], [29]. The reason behind WOA's excellent performance is because of its searching mechanism, where it imitates the bubble-net feeding method of humpback whales, as shown in Fig. 1.

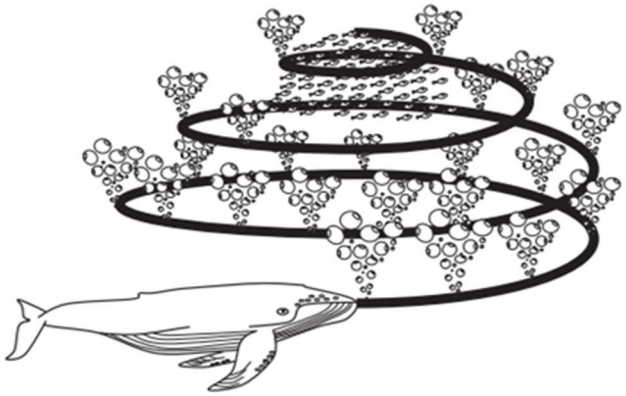

Fig. 1 The bubble-net feeding method of humpback whales [15]

1) Encircling prey: In this phase, the whales encircle the prey. Since the optimal solution of the prey is not yet identified, the current best position is assumed to be the prey. Once the best position is known, the whales update their positions toward the current best position over iterations. The scenario is represented by Eq. (1) and Eq. (2), where t denotes the current iteration, $\vec{X}$ is the position vector, $\overrightarrow{X*}$ represents the position vector of the best position obtained at the moment, while $||$ and $\cdot$ indicate the absolute value and dot product operator, respectively. The value of $\overrightarrow{X*}$ would be updated if a better position is found at each iteration. The distance between the current whale and the best position is denoted by $\vec{D}$. The coefficient vectors $\vec{A}$ and $\vec{C}$ are shown in Eq. (3) and Eq. (4) where $\vec{r}$ is a random vector between 0 and 1, while $\vec{a}$ is a control parameter that decreases linearly from 2 to 0 using Eq. (5), where $MaxIter$ indicates the maximum iteration.

$$\vec{D} = |\vec{C} \cdot \overrightarrow{X*}(t) - \vec{X}(t)| \quad (1)$$

$$\vec{X}(t+1) = \overrightarrow{X*}(t) - \vec{A} \cdot \vec{D} \quad (2)$$

$$\vec{A} = 2\vec{a} \cdot \vec{r} - \vec{a} \quad (3)$$

$$\vec{C} = 2 \cdot \vec{r} \quad (4)$$

$$a = 2 - t\frac{2}{MaxIter} \quad (5)$$

2) Bubble-net attacking (exploitation): In this phase, the whales perform bubble-net attacking by two maneuvers: shrinking the circle and moving along the spiral-shaped route. The first maneuver is achieved by Eq. (2) while the second maneuver is achieved by Eq. (6) where b indicates a constant to create the shape of the logarithmic spiral, l denotes a random number between –1 and 1, and $\overrightarrow{D'}$ is computed from $\overrightarrow{D'} = |\overrightarrow{X*}(t) - \vec{X}(t)|$, which means the distance between the i[th] whale to the prey. Since the whales need to carry out these two maneuvers at the same time, hence, p is used as a probability to decide which maneuver to perform. This scenario is presented in Eq. (7). If $p < 0.5$, the whales shrink the circle, otherwise they move along the spiral-shaped route.

$$\vec{X}(t+1) = \overrightarrow{D'} \cdot e^{bl} \cdot cos(2\pi l) + \overrightarrow{X*}(t) \quad (6)$$

$$\vec{X}(t+1) = \begin{cases} \overrightarrow{X*}(t) - \vec{A} \cdot \vec{D} & if\ p < 0.5 \\ \overrightarrow{D'} \cdot e^{bl} \cdot cos(2\pi l) + \overrightarrow{X*}(t) & if\ p \geq 0.5 \end{cases} \quad (7)$$

3) Searching for prey (exploration): In this phase, whales look for the potential position randomly based on their positions among one another. It is unlike the exploitation phase because the position of a whale is updated using a randomly selected whale, instead of using the best position discovered. This scenario is represented in Eq. (8) and Eq. (9), where $\overrightarrow{X_{rand}}$ indicates a random whale's position selected from the population.

$$\vec{D} = |\vec{C} \cdot \overrightarrow{X_{rand}} - \vec{X}| \quad (8)$$

$$\vec{X}(t+1) = \overrightarrow{X_{rand}} - \vec{A} \cdot \vec{D} \quad (9)$$

The pseudocode of WOA is shown in Fig. 2.

```
Initialize whales' population Xi (i = 1, 2, ..., n)
Compute each search agent's fitness
X*=the best search agent
While t smaller than MaxIter
    for each search agent
        Update a, A, C, l, and p with Eq. (5), (3), (4), [-1,1], [0,1]
        if p smaller than 0.5
            if |A| smaller than 1
                Update the current search agent's position by Eq. (2)
            else
                Choose a search agent by random (Xrand)
                Update the current search agent's position by Eq. (9)
            end if
        else
            Update the current search agent's position by Eq. (6)
        end if
    end for
    Fix the search agent's position if it outstrips the search space
    Compute each search agent's fitness
    Update X* if a fitter solution is found
    t=t+1
end while
return X*
```

Fig. 2 Pseudocode of WOA

B. Grey Wolf Optimizer

Grey Wolf Optimizer (GWO) is a nature-inspired metaheuristic algorithm introduced by Mirjalili *et al*. in 2014 for solving optimization issues [14]. It has shown great performance in electromagnetics [17] and feature selection [22], [23]. GWO is well-perform due to the grey wolves' hunting mechanism that involves a special social hierarchy.

1) Social hierarchy: The algorithm of GWO is inspired by the grey wolves' social hierarchy in group hunting. The hierarchy consists of four levels which are alpha (α), beta (β), delta (δ), and omega (ω) from top to bottom as illustrated in Fig. 3.

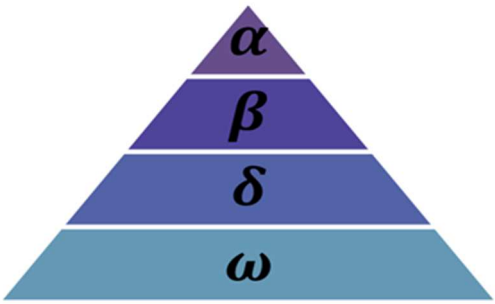

Fig. 3 Grey wolves' social hierarchy [14]

The alpha wolves are the highest in the social hierarchy, and they lead the other three levels below them. Being the second highest in the hierarchy, beta wolves obey alpha wolves and dominate delta and omega wolves. Similarly, delta wolves are dominated by the first two levels of wolves and dominate the bottom level of wolves. The last in the hierarchy is called the omega wolves, and the top three levels of wolves dominate them. The social hierarchy concept is transformed into the algorithm of GWO where alpha, beta, and delta are categorized as the fittest, second fittest, and third fittest solutions, respectively. The solutions other than the top three fittest ones are called omega.

2) Encircling prey: In this phase, the wolves encircle the prey (best position). The scenario is represented by Eq. (10) and Eq. (11) where t represents current iteration, $\vec{X}$ denotes the position vector of a grey wolf, $\vec{X}_p$ represents the position vector of the fittest solution obtained so far, and $\cdot$ is the dot product operator. $\vec{X}_p$ will be updated in each iteration once better position is found. Parameter $\vec{D}$ denotes the distance between the current wolf and the as-is best position. The coefficient vectors $\vec{A}$ and $\vec{C}$ are presented in Eq. (12) and Eq. (13), respectively, where $\vec{r}_1$ and $\vec{r}_2$ are random numbers between 0 and 1, whereas $\vec{a}$ is a control parameter that decreases linearly from 2 to 0. The decrease in $\vec{a}$ is done using Eq. (14), where $MaxIter$ indicates the maximum iteration, similar to WOA.

$$\vec{D} = |\vec{C} \cdot \vec{X}_p(t) - \vec{X}(t)| \tag{10}$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D} \tag{11}$$

$$\vec{A} = 2\vec{a} \cdot \vec{r}_1 - \vec{a} \tag{12}$$

$$\vec{C} = 2 \cdot \vec{r}_2 \tag{13}$$

$$a = 2 - t\frac{2}{MaxIter} \tag{14}$$

3) Hunting: The hunting mechanism of GWO depends on alpha, beta, and delta wolves, as shown in Eq. (15) and Eq. (16). It is worth mentioning that the top three fittest solutions are assumed to be the best position since the optimal solution is unknown. Hence, GWO uses the mean position from alpha, beta, and delta wolves to update each wolf's position based on Eq. (17).

$$\begin{aligned}\overrightarrow{D_\alpha} = |\overrightarrow{C_1} \cdot \overrightarrow{X_\alpha} - \vec{X}|, \overrightarrow{D_\beta} &= |\overrightarrow{C_2} \cdot \overrightarrow{X_\beta} - \vec{X}|, \overrightarrow{D_\delta} \\ &= |\overrightarrow{C_3} \cdot \overrightarrow{X_\delta} - \vec{X}|\end{aligned} \tag{15}$$

$$\begin{aligned}\overrightarrow{X_1} = \overrightarrow{X_\alpha} - \overrightarrow{A_1} \cdot \overrightarrow{D_\alpha}, \overrightarrow{X_2} &= \overrightarrow{X_\beta} - \overrightarrow{A_2} \cdot \overrightarrow{D_\beta}, \overrightarrow{X_3} \\ &= \overrightarrow{X_\delta} - \overrightarrow{A_3} \cdot \overrightarrow{D_\delta}\end{aligned} \tag{16}$$

$$\vec{X}(t+1) = \frac{\overrightarrow{X_1} + \overrightarrow{X_2} + \overrightarrow{X_3}}{3} \tag{17}$$

4) Attacking prey (exploitation): The process of attacking prey is also known as the local search, where grey wolves hunt the prey as it stops moving. The value of $\vec{A}$ is affected by $\vec{a}$ whereby $\vec{A}$ falls randomly between the range of $-2a$ to $2a$ based on Eq. (12). If $|\vec{A}| < 1$, the value of $\vec{A}$ is between –1 to 1, thus, the wolf attacks the prey because the wolf's next position would be between its current position and the prey's position.

5) Searching for prey (exploration): The search process is also known as the global search, where all the top three levels of wolves' position are used to look for the best solution. The global search is the opposite of the local search. Each wolf looks for prey separately and then converges toward the best solution found. If $|\vec{A}| > 1$, the value of $\vec{A}$ is smaller than –1 or larger than 1, thus, the wolf stays away from the current prey position to find a better solution. Therefore, this vector manages the balance of exploration and exploitation. Besides $\vec{A}$, the $\vec{C}$ vector also determines the exploration. The value of $\vec{C}$ is a random number between 0 to 2, and it randomly weights ($C > 1$) and unweights ($C < 1$) the effect of prey by the distance. This concept allows the algorithm to exhibit random behavior during optimization, encouraging exploration and eradicating local optima.

The pseudocode of GWO is shown in Fig. 4.

```
Initialize wolves' population Xi (i = 1, 2, ..., n)
Initialize a, A, and C with Eq. (14), (12), (13), respectively
Compute each search agent's fitness
Xα=the best search agent
Xβ =the second-best search agent
Xδ =the third-best search agent
While t smaller than MaxIter
    for each search agent
        Update the current search agent's position by Eq. (17)
    end for
    Update a, A, and C with Eq. (14), (12), (13), respectively
    Compute each search agent's fitness
    Update Xα, Xβ, and Xδ
    t=t+1
end while
return Xα
```

Fig. 4 Pseudocode of GWO

C. Proposed Inversed Control Parameter

Based on the literature, mWOA was designed for filter-based feature selection to improve the convergence speed while performing a 50% of feature reduction on four medical HDDs [24]. However, its performance has not yet been tested in a wrapper-based feature selection method nor evaluated on datasets of different dimensionality. Hence, this study proposed two wrapper-based feature selection methods using modified WOA (mWOA) and modified GWO (mGWO) to improve the convergence speed by inversing the control parameter.

In the original WOA and GWO, the control parameter is defined by Eq. (5) and Eq. (14), respectively. The values of a in WOA and GWO decrease linearly from 2 to 0 and affect both global and local search mechanisms. This is because the values of the coefficient vector, $\vec{A}$, are contributed by a, while $\vec{A}$ determines the updated position of search agents in both methods. The values of the control parameter, a, and the coefficient vector $\vec{A}$, are shown in Fig. 5.

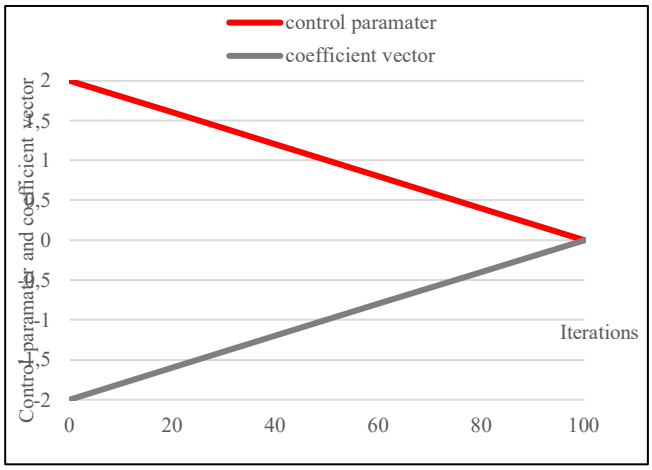

Fig. 5 Values of the control parameter, a, and coefficient vector $\vec{A}$ in original WOA and GWO

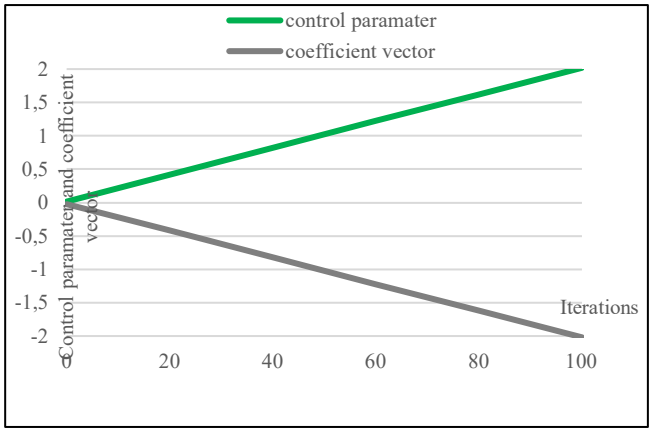

Fig. 6 Values of the control parameter, a, and coefficient vector $\vec{A}$ in proposed mWOA and mGWO

Meanwhile, the decreasing trend of a decreases the distance and increases the possibility of changing positions, which covers less search space at the initial phase of the iterations, resulting in a slower convergence speed [23]. Hence, the situation had to be changed the other way around to enable faster convergence speed [24]. Therefore, the linearly decreasing control parameter in WOA and GWO was inversed to be linearly increasing with Eq. (18) for the proposed mWOA and mGWO. With the new control parameter formula, the value of a linearly increases from 0 to 2, as depicted in Fig. 6.

$$a = \frac{2t + 2}{MaxIter} \tag{18}$$

TABLE I
COMPARISON OF THE ORIGINAL AND PROPOSED METHODS

	Original WOA and GWO	Proposed mWOA and mGWO
Value of a	Decreasing	Increasing
Distance between search agents	Decreasing	Increasing
Position-changing possibility	Increasing	Decreasing
Search space covered	Less at first, more later	More at first, less later
Resulting Convergence Speed	Slower	Faster

Table 1 shows the comparison of the original and proposed methods. By using the inversed control parameter from Eq. (18), the increasing trend of a increases the distance and decreases the possibility of changing positions, which covers more search space at the initial phase of the iterations, a faster convergence speed in the proposed mWOA and mGWO.

The inversed control parameter in the proposed mWOA and mGWO is expected to improve the convergence speed of WOA and GWO in wrapper-based feature selection. With better convergence speed reflecting the algorithm performance, the search agents are most likely to excel in selecting only the most important features in a dataset that truly contribute to the classification accuracy. Therefore, in the following subsection, the effectiveness of the inversed control parameter for both mWOA and mGWO against their original forms of GWO and WOA, was tested in experiments.

D. Experimental Setup

The experiments were implemented using MATLAB version R2017b. A total of 12 benchmark datasets from UCI [30] with different dimensionality were used. The datasets are sorted in descending order based on dimensionality (*instance* × *feature*), as shown in Table 2. There are four HDDs with more than 2,000 features in the datasets, namely, SMK_CAN_187, GLI_85, CNS, and Colon, where the rest of the datasets were non-HDDs.

TABLE II
LIST OF DATASETS EMPLOYED IN THIS STUDY

Datasets	Instances	Features	Dimensionality
SMK_CAN_187	187	19993	3738691
GLI_85	85	22283	1894055
CNS	60	7129	427740
Colon	62	2000	124000
Wdbc	569	30	17070
Ionosphere	351	34	11934
Yeast	1484	8	11872
SPECT	267	22	5874
Parkinson's	195	22	4290
Wine	178	13	2314
Zoo	101	16	1616
Iris	150	4	600

A wrapper-based feature selection approach with the kNN classifier was used, where the k was fixed to 5. Cross-validation was implemented in this study, where each dataset was randomly partitioned for hold-out in N validation, where N denotes the number of instances in a dataset. The partition randomly divided N instances into two subsets: 20% for testing sets and 80% for training sets. The same setting was used for each experiment to avoid bias in results. The parameters for population maximum iteration were 10 and 100, respectively. The machine that was used to run the experiments was Intel Core i7-10750H CPU @ 2.60GHz with 32GB RAM. The experiments were conducted for 10 runs with two pairs of algorithms, which include WOA and mWOA, as well as GWO and mGWO. In the next section, both the proposed mWOA and mGWO were compared against their original forms with the following criteria:

- The number of selected features in best, average, and worst cases: the best number of selected features was obtained by the minimum value in 10 runs, while the maximum value was obtained by the worst number of selected features in 10 runs. The average number of selected features was computed by the mean of selected features in all ten runs. The feature reduction rate can also be computed using Eq. (19).

$$Feature\ reduction\ rate = \left(1 - \frac{number\ of\ selected\ features}{original\ feature\ size}\right) \times 100\% \quad (19)$$

- The classification accuracy in best, average, and worst cases was obtained using the selected features on the test dataset. The maximum value selected the best accuracy in 10 runs, while the worst accuracy was selected by the minimum value in 10 runs. The average accuracy was computed by the mean of accuracies in all ten runs. The accuracy of each experiment was calculated using Eq. (20) [10], where TP and TN indicate the numbers of positive and negative instances that are correctly classified, whereas FP and FN represent the numbers of positive and negative instances that are incorrectly classified.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (20)$$

III. Results And Discussion

This section presents the experimental results of the proposed mWOA and mGWO against their original forms and discusses the effectiveness of the inversed control parameter based on the number of selected features and classification accuracy.

A. The number of selected features

The wrapper-based feature selection experiments utilized 12 datasets in Table 2 for 10 runs, and the number of selected features vary across each test case. Table 3 presents the number of selected features by WOA and mWOA in their best, average, and worst cases. Based on the experimental results, mWOA performed better than WOA in 6, 6, and 8 out of 12 datasets in the best, average, and worst test cases, respectively. Also, it achieved the same performance as WOA in 5, 3, and 2 out of 12 datasets in the best, average, and worst test cases, respectively. It indicates that mWOA could select fewer features in at least 50-91.67% of datasets in the best case, 50-75% of datasets in average cases, and 66.67-83.33% of datasets in the worst case.

TABLE III
COMPARISON OF THE NO. OF SELECTED FEATURES IN BEST, AVERAGE, AND WORST CASES FOR WOA AND MWOA

Datasets	No. of selected features (features reduction rate %)					
	Best		Average		Worst	
	WOA	mWOA	WOA	mWOA	WOA	mWOA
SMK_CAN_187	13 (99.93)	**8 (99.96)** *	297 (98.51)	548 (97.26)	640 (96.80)	4106 (79.46)
GLI_85	15 (99.93)	**3 (99.99)** *	1071 (95.19)	**792 (96.45)** *	8821 (60.41)	**2868 (87.13)** *
CNS	5 (99.93)	**2 (99.97)** *	197 (97.24)	**110 (98.46)** *	1356 (80.98)	**606 (91.50)** *
Colon	8 (99.60)	**6 (99.70)** *	125 (93.75)	130 (93.50)	477 (76.15)	**466 (76.70)** *
Wdbc	3 (90.00)	**2 (93.33)** *	8 (73.33)	**4 (86.67)** *	12 (60.00)	**8 (73.33)** *
Ionosphere	3 (91.18)	**3 (91.18)** #	5 (85.29)	**4 (88.24)** *	7 (79.41)	**7 (79.41)** #
Yeast	5 (37.50)	**5 (37.50)** #	7 (12.50)	**7 (12.50)** #	8 (0.00)	**8 (0.00)** #
SPECT	6 (72.73)	**1 (95.45)** *	12 (45.45)	**9 (59.09)** *	16 (27.27)	**15 (31.82)** *
Parkinson's	2 (90.91)	**2 (90.91)** #	6 (72.72)	**4 (81.82)** *	11 (50.00)	**8 (63.64)** *
Wine	2 (84.62)	**2 (84.62)** #	4 (69.23)	6 (53.85)	9 (30.77)	**8 (38.46)** *
Zoo	3 (81.25)	4 (75.00)	7 (56.25)	**7 (56.25)** #	12 (25.00)	**9 (43.75)** *
Iris	1 (75.00)	**1 (75.00)** #	2 (50.00)	**2 (50.00)** #	3 (25.00)	4 (0.00)
Mean	6 (85.21)	3 (86.88)	145 (70.79)	135 (72.84)	948 (50.98)	676 (55.43)

Note: Bold value with an asterisk (*) indicates that the proposed mWOA outperformed WOA, while a hash sign (#) indicates they achieved the same performance.

Based on Table 3, in the best case, mWOA outperformed WOA in all HDDs, which includes SMK_CAN_187, GLI_85, CNS, and Colon. In GLI_85 dataset, mWOA achieved the highest feature reduction rate at 99.99% by only selecting 3 from 22,283 features. Besides, mWOA also showed the best improvement of non-HDDs such as in SPECT dataset, as mWOA reduced 95.45% of features by only choosing 1 from 22 features. It is equivalent to a 22.73% improvement achieved by mWOA compared to WOA (72.73%).

In average cases, mWOA outperformed WOA in two HDDs, which are GLI_85 and CNS. All four HDDs were successfully reduced by more than 93.50% of features using mWOA. Specifically, SMK_CAN_187 reduced to 548 from 19,993 features, GLI_85 reduced to 792 from 22,283 features, CNS reduced to 110 from 7,129 features, and Colon reduced to 130 from 2,000 features. For non-HDDs, mWOA managed to reduce 86.67% of features in the Wdbc dataset, that is, 13.33% better than WOA, which reduced 73.33% of features.

Even in the worst case, mWOA showed its strength in selecting fewer features than WOA in both HDDs and non-HDDs. In the GLI_85 dataset, mWOA successfully selected 2,868 features resulting in feature reduction rates of 87.13% as compared to WOA with 60.41%. The improvement that mWOA achieved in this HDD is 26.72%. As for non-HDDs, mWOA managed to reduce 43.75% of features as compared to WOA with 25% of feature reduction rate, indicating that mWOA was 18.75% better than WOA.

Table 4 shows the number of selected features by GWO and mGWO in their best, average, and worst cases, respectively. The experimental results show that the mGWO outperformed GWO in 2 datasets in the worst case and achieved the same performance as GWO in 6 datasets in the best case, 3 in average cases, and 1 in the worst case. Overall, the performance of mGWO was not significant as compared to GWO.

TABLE IV
COMPARISON OF THE NO. OF SELECTED FEATURES IN BEST, AVERAGE, AND WORST CASES FOR GWO AND MGWO

Datasets	No. of selected features (features reduction rate %)					
	Best		Average		Worst	
	GWO	mGWO	GWO	mGWO	GWO	mGWO
SMK_CAN_187	2944 (85.27)	3808 (80.95)	3302 (83.48)	5778 (71.10)	3801 (80.99)	8619 (56.89)
GLI_85	2706 (87.86)	3077 (86.19)	3282 (85.27)	3799 (82.95)	4706 (78.88)	**4553 (79.57) ***
CNS	820 (88.50)	1393 (80.46)	1006 (85.89)	1610 (77.42)	1138 (84.04)	1971 (72.35)
Colon	159 (92.05)	234 (88.30)	184 (90.80)	310 (84.50)	215 (89.25)	431 (78.45)
Wdbc	2 (93.33)	**2 (93.33) #**	4 (86.67)	6 (80.00)	7 (76.67)	10 (66.67)
Ionosphere	3 (91.18)	4 (88.24)	5 (85.29)	8 (76.47)	8 (76.47)	12 (64.71)
Yeast	5 (37.5)	**5 (37.5) #**	6 (25.00)	**6 (25.00) #**	8 (0)	**8 (0) #**
SPECT	3 (86.36)	5 (77.27)	6 (72.73)	7 (68.18)	8 (63.64)	9 (59.09)
Parkinson's	2 (90.91)	**2 (90.91) #**	2 (90.91)	3 (86.36)	4 (81.82)	5 (77.27)
Wine	2 (84.62)	**2 (84.62) #**	4 (69.23)	**4 (69.23) #**	5 (61.54)	7 (46.15)
Zoo	4 (75.00)	**4 (75.00) #**	6 (62.50)	**6 (62.50) #**	8 (50.00)	**7 (56.25) ***
Iris	1 (75.00)	**1 (75.00) #**	1 (75.00)	2 (50.00)	3 (25.00)	4 (0.00)
Mean	554 (82.30)	711 (79.81)	651 (76.06)	962 (69.48)	826 (64.02)	1303 (54.78)

Note: Bold value with an asterisk (*) indicates that the proposed mGWO outperformed GWO, while a hash sign (#) indicates they achieved the same performance.

As shown by the best case results in Table 4, mGWO did not outrun GWO in all four HDDs. However, mGWO was still able to achieve high feature reduction rates on HDDs where they were reduced by at least 80% of features. For example, SMK_CAN_187 was reduced to 3,808 from 19,993 features, GLI_85 to 3,077 from 22,283 features, and CNS to 1,393 from 7,129 features, and the selected features for Colon was 234 out of 2,000 features. As for the non-HDDs, mGWO remained the same number of selected features as GWO in most datasets. Nonetheless, mGWO and GWO both reduced a maximum of 93.33% and 90.91% of features in Wdbc and Parkinson's datasets, respectively, indicating that both GWO and mGWO were capable of dealing with HDDs and non-HDDs.

While comparing the performance of mGWO and GWO in selecting fewer features in the average cases, mGWO did not show its strength in HDDs. As for non-HDDs, mGWO mostly showed values that remained the same as GWO. It can be seen that mGWO's best performance was in non-HDDs, especially the Parkinson's dataset, where it was reduced to 86.36% by selecting only 3 from 22 features. However, GWO selected 1 feature fewer than mGWO in the same dataset.

Further evaluating the worst case shown in Table 4, mGWO was weaker than GWO. There were only two datasets where mGWO outperformed GWO, which include an HDD and a non-HDD, namely, GLI_85 and Zoo. In the GLI_85 dataset, mGWO selected 4,553 features with a 79.57% of feature reduction rate while GWO only reduced 78.88%, suggesting a slight improvement of 0.69% made by mGWO. The improvement of mGWO in the non-HDD dataset, Zoo was much higher than GWO, where mGWO achieved a higher ability to select 6.25% fewer features than GWO.

To summarize the feature selection performance, by taking the mean from all datasets, the proposed mWOA showed greater performance in selecting 1.67%, 2.05%, and 4.45% fewer features than WOA in its worst, average, and best cases. Also, mWOA improved the feature reduction rate by 22.73% better than WOA on SPECT dataset in the best case, 13.33% better than WOA on Wdbc dataset in the average case, and 26.72% better than WOA on GLI_85 HDD even in its worst case. Besides, the proposed mGWO's feature selection performance was less significant because it was merely maintaining similar results as the original GWO without major improvement. The following subsection presents the experimental results of the proposed mWOA and mGWO against WOA and GWO based on classification accuracy.

B. Classification accuracy

The classification accuracy obtained by the selected features was evaluated using the kNN classifier. The best, average, and worst results obtained from mWOA against WOA were tabulated in Table 5. Based on the results, mWOA outperformed WOA in 2, 7, and 4 out of 12 datasets in the best, average, and worst test cases, respectively. mWOA also performed equally as WOA in 9, 1, and 5 out of 12 datasets. It suggested that mWOA was able to produce higher accuracy than WOA in at least 16.67-91.67% of datasets in the best case, 58.33-66.67% of datasets in average cases, and 33.33-41.67% of datasets in the worst case.

TABLE V
COMPARISON OF THE CLASSIFICATION ACCURACY IN BEST, AVERAGE, AND WORST CASES FOR WOA AND MWOA

Datasets	Classification Accuracy (%)					
	Best		Average		Worst	
	WOA	mWOA	WOA	mWOA	WOA	mWOA
SMK_CAN_187	83.78	81.08	78.11	77.57	72.97	**72.97 #**
GLI_85	100	**100 #**	96.47	**96.47 #**	88.24	82.35
CNS	100	**100 #**	89.17	87.50	75.00	**83.33 ***
Colon	100	**100 #**	96.67	**98.33 ***	91.67	**91.67 #**
Wdbc	98.23	**98.23 #**	95.75	94.96	91.15	**92.04 ***
Ionosphere	97.14	**97.14 #**	92.29	**92.71 ***	90.00	**90.00 #**
Yeast	60.81	**60.81 #**	57.20	**57.64 ***	54.05	50.68
SPECT	88.68	**90.57 ***	86.42	**86.60 ***	81.13	79.25
Parkinson's	94.87	**94.87 #**	92.05	**92.31 ***	87.18	**87.18 #**
Wine	97.14	**100 ***	94.57	**96.29 ***	82.86	**94.29 ***
Zoo	100	**100 #**	97.00	**97.50 ***	85.00	**90.00 ***
Iris	100	**100 #**	98.00	97.67	93.33	**93.33 #**
Mean	93.39	93.56	89.48	89.63	82.72	83.92

Note: Bold value with an asterisk (*) indicates that the proposed mWOA outperformed WOA, while a hash sign (#) indicates they achieved the same performance.

It can be observed from the best case that mWOA had similar results as WOA in general. In the four HDDs, mWOA produced comparable results to WOA where GLI_85, CNS, and Colon showed 100% accuracy. However, mWOA was proven to have better performance in non-HDDs as compared to WOA. Specifically, mWOA obtained 90.57% accuracy, 1.89% higher than WOA (88.68%) in the SPECT dataset. Also, the highest performance of mWOA can be found in the Wine dataset where mWOA achieved 100% of accuracy, which is 2.86% higher than WOA.

Moreover, based on the results in the average case, mWOA outperformed WOA in most non-HDDs, including Ionosphere, Yeast, SPECT, Parkinson's, Wine, and Zoo datasets. The fact that mWOA also showed the highest improvement in the Wine dataset by obtaining 1.72% better than WOA, has proven the ability of mWOA to increase the accuracy in non-HDDs. Although mWOA did not surpass WOA in Wdbc and Iris datasets, mWOA still managed to generate high accuracies at 94.96% and 97.67% in the Wdbc and Iris datasets, respectively. Besides, mWOA triumphed over WOA with a significant difference in Colon dataset. mWOA and WOA attained 98.33% and 96.67% of accuracy respectively, indicating that mWOA was able to select relevant features that contributed to 1.66% higher classification accuracy than WOA in HDD.

In addition, even in the worst-case scenario, mWOA was able to display significant improvements across datasets with different dimensionality. For instance, in the CNS dataset, mWOA achieved 83.33% accuracy, equivalent to 8.33% higher than WOA (75%). Not only showing its ability in generating high accuracy in HDD, mWOA also attained an 11.43% improvement in a non-HDD when mWOA and WOA achieved 94.29% and 82.86% of accuracy, respectively in the Wine dataset. Another significant result of mWOA in non-HDD can be found in the Zoo dataset with 90% of classification accuracy, that is 5% better than WOA.

TABLE VI
CO COMPARISON OF THE CLASSIFICATION ACCURACY IN BEST, AVERAGE, AND WORST CASES FOR GWO AND MGWO

Datasets	Classification Accuracy (%)					
	Best		Average		Worst	
	GWO	mGWO	GWO	mGWO	GWO	mGWO
SMK_CAN_187	86.49	83.78	80.0	77.57	75.68	72.97
GLI_85	100	**100 #**	97.06	**97.06 #**	88.24	**88.24 #**
CNS	100	**100 #**	92.50	90.00	83.33	75.00
Colon	100	**100 #**	99.17	98.33	91.67	**91.67 #**
Wdbc	98.23	**98.23 #**	95.93	**95.93 #**	92.04	**92.04 #**
Ionosphere	97.14	**98.57 ***	95.00	**95.00 #**	92.86	**92.86 #**
Yeast	60.81	60.47	56.96	**57.84 ***	54.05	**55.07 ***
SPECT	88.68	**88.68 #**	86.79	**87.17 ***	81.13	**81.13 #**
Parkinson's	94.87	**97.44 ***	91.79	**93.08 ***	89.74	87.18
Wine	100	**100 #**	97.71	**98.00 ***	94.29	**94.29 #**
Zoo	100	**100 #**	98.00	**98.00 #**	95.00	**95.00 #**
Iris	100	**100 #**	97.67	**98.33 ***	93.33	**93.33 #**
Mean	93.85	93.93	90.72	90.53	85.95	84.90

Note: Bold value with an asterisk (*) indicates that the proposed mGWO outperformed GWO, while a hash sign (#) indicates they achieved the same performance.

Furthermore, Table 6 presents the classification accuracy obtained by GWO and mGWO in best, average, and worst cases. Based on the experimental results, mGWO surpassed GWO on 2 datasets in the best case, 5 in the average case, and 1 in the worst case. Additionally, mGWO maintained the same performance as GWO on 8, 4, and 8 datasets in the best, average, and worst case, respectively. It means that mGWO performed better than GWO in at least 16.67-83.33% of datasets in the best case, 41.67-75% in the average case, and 8.33-75% in the worst case.

It can be seen that the best-case classification accuracies obtained by mGWO and GWO on every dataset in Table 6

were mostly higher than 80% of accuracy. However, for the Yeast dataset, mGWO and GWO only generated 60.47% and 60.81% accuracies, respectively. This might be because the Yeast dataset may possess a complex or uncleaned data structure that somehow degraded the classification process. Besides, mGWO showed better classification accuracy on the rest of the HDDs and non-HDDs in the best case. For instance, mGWO achieved 97.44% of accuracy whereas GWO obtained 94.87% of accuracy, indicating that mGWO was 2.57% better in the Parkinson's dataset. In HDDs, mGWO was able to attain 100% of accuracy in GLI_85, CNS, and Colon datasets, the same as GWO.

As for the average case, the classification accuracy of the four HDDs obtained by mGWO was either weaker or the same as GWO. Nevertheless, mGWO outperformed GWO mostly in non-HDDs. Similar to the best-case results, the highest improvement made by mGWO against GWO in the average case can be found in the Parkinson's dataset, where mGWO achieved 93.08% of accuracy, which was 1.29% higher than GWO. In the worst case, mGWO only maintained the performance as GWO in most datasets. The accuracy attained by mGWO in HDDs was weaker than GWO. There was only 1.02% improvement of mGWO in the Yeast dataset compared to GWO.

In a nutshell, mWOA was able to achieve 0.17%, 0.15%, and 1.21% higher accuracy than WOA in the best, average, and worst cases, by taking the mean from all datasets. In non-HDDs like the Wine dataset, mWOA achieved 2.86% better than WOA in the best case, 1.66% better than WOA on the Colon HDD in the average case, and 8.33% better than WOA on the CNS HDD in the worst case. As for the proposed mGWO, its performance in classification accuracy was mostly maintaining comparable results as the original GWO. However, mGWO did show some improvement in certain non-HDDs with slightly better accuracy than GWO in the average case.The following section concludes the comparative study and suggests future work related to the extension of this work.

IV. Conclusion

Previously, the authors proposed a filter-based feature selection method with a modified WOA (mWOA) to improve the convergence speed and achieve better accuracy performance than WOA. However, GWO was also found to be having the same issue of slow convergence speed. Therefore, as an extension of the previous work, the authors began with this comparative study by proposing wrapper-based feature selection methods with mWOA and modified GWO (mGWO) using an inversed control parameter, a. The inversed control parameter was expected to allow search agents to cover more search space in the early iterations to improve the convergence speed by obtaining the optimal solution faster. Twelve datasets with different dimensionality from UCI were adopted in the experiments, and the kNN classifier was used to evaluate the selected features. The performance of the proposed methods was evaluated against their original algorithms in terms of the number of selected features and classification accuracy.

Based on the experimental results, it can be concluded that the strength of the proposed mWOA and mGWO was shown differently. Although GWO and WOA share similarities in their algorithms and both have the issue of slow convergence, using the same method is not applicable for both. In terms of feature reduction rate, mWOA obtained 1.67%, 2.05%, and 4.45% better results than WOA in the best, average, and worst cases, respectively. However, mGWO did not outperform GWO in most cases. As for the classification accuracy, mWOA outperformed WOA with 0.17%, 0.15%, and 1.2% better results in the best, average, and worst cases, respectively. Nonetheless, mGWO outperformed 0.08% in classification accuracy than GWO in the best case, yet, no improvements in the average and worst cases. It was proven that the proposed inversed control parameter is only effective in mWOA, but less effective in mGWO, when comparing them against their original algorithms.

In the future, the execution time of feature selection could be considered as one of the criteria to evaluate the proposed methods. Besides, since the inversed control parameter showed more significant improvement on mWOA, thus, hybridizing mWOA with other recent well-perform metaheuristic algorithms to further investigate the performance would be an interesting study. Some recent metaheuristic algorithms that could be hybridized with mWOA include Manta Ray Foraging Optimization and Harris Hawks Optimization. Lastly, other classifiers could be wrapped with mGWO to select relevant features that produce better classification accuracy in HDDs.

Acknowledgment

This research was supported by Ministry of Higher Education (MOHE) through Fundamental Research Grant Scheme (FRGS/1/2018/ICT02/UTHM/02/6).

References

[1] O. Duncan and T. Sherer, "Feature Selection (Data Mining)," *Microsoft*, 2018. [Online]. Available: https://docs.microsoft.com/en-us/analysis-services/data-mining/feature-selection-data-mining?view=asallproducts-allversions. [Accessed: 01-May-2021].

[2] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, *Feature Selection for High-Dimensional Data*. Springer International Publishing, 2015.

[3] A. Bommert, X. Sun, B. Bischl, J. Rahnenführer, and M. Lang, "Benchmark for filter methods for feature selection in high-dimensional classification data," *Comput. Stat. Data Anal.*, vol. 143, p. 106839, 2020.

[4] B. Zhang and P. Cao, "Classification of high dimensional biomedical data based on feature selection using redundant removal," *PLoS One*, vol. 14, no. 4, pp. 1–19, 2019.

[5] A. Veeraswamy and A. M. Babu, "Classification of High Dimensional Data Using Filtration Attribute Evaluation Feature Selection Method of Data mining," *4th Int. Conf. Electr. Electron. Commun. Comput. Technol. Optim. Tech. ICEECCOT 2019*, pp. 8–12, 2019.

[6] K. S. Adewole *et al.*, "Hybrid Feature Selection Framework For Sentiment Analysis On Large Corpora," *Jordanian J. Comput. Inf. Technol.*, vol. 07, no. 02, pp. 15–33, 2021.

[7] Q. Al-Tashi, S. J. Abdulkadir, H. M. Rais, S. Mirjalili, and H. Alhussian, "Approaches to Multi-Objective Feature Selection: A Systematic Literature Review," *IEEE Access*, vol. 8, pp. 125076–125096, 2020.

[8] R. Alazaidah, M. A. Almaiah, and M. Al-Luwaici, "Associative Classification In Multi-label Classification: An Investigative Study," *Jordanian J. Comput. Inf. Technol.*, vol. 7, no. 2, pp. 166–179, 2021.

[9] Y. Bouchlaghem, Y. Akhiat, and S. Amjad, "Feature Selection: A Review and Comparative Study," *E3S Web Conf.*, vol. 351, p. 01046, 2022.

[10] H. Nematzadeh, R. Enayatifar, M. Mahmud, and E. Akbari, "Frequency based feature selection method using whale algorithm," *Genomics*, vol. 111, no. 6, pp. 1946–1955, 2019.

[11] H. M. Mohammed, S. U. Umar, and T. A. Rashid, "A systematic and meta-analysis survey of whale optimization algorithm," *Comput. Intell. Neurosci.*, vol. 2019, 2019.
[12] W. Zhao, Z. Zhang, and L. Wang, "Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications," *Eng. Appl. Artif. Intell.*, vol. 87, no. September 2019, p. 103300, 2020.
[13] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," *Futur. Gener. Comput. Syst.*, vol. 97, pp. 849–872, 2019.
[14] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, 2014.
[15] S. Mirjalili and A. Lewis, "The Whale Optimization Algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, 2016.
[16] R. K. Saidala and N. Devarakonda, "Improved whale optimization algorithm case study: Clinical data of anaemic pregnant woman," *Adv. Intell. Syst. Comput.*, vol. 542, pp. 271–281, 2018.
[17] X. Li and K. M. Luk, "The Grey Wolf Optimizer and Its Applications in Electromagnetics," *IEEE Trans. Antennas Propag.*, vol. 68, no. 3, pp. 2186–2197, 2020.
[18] B. Sony, A. Chakravarti, and M. M. Reddy, "Traffic congestion detection using whale optimization algorithm and multi-support vector machine," *Int. J. Recent Technol. Eng.*, vol. 7, no. 6C2, pp. 589–593, 2019.
[19] M. Mafarja and S. Mirjalili, "Whale optimization approaches for wrapper feature selection," *Appl. Soft Comput.*, vol. 62, pp. 441–453, 2018.
[20] M. M. Mafarja and S. Mirjalili, "Hybrid Whale Optimization Algorithm with simulated annealing for feature selection," *Neurocomputing*, vol. 260, pp. 302–312, 2017.
[21] K. K. Ghosh, R. Guha, S. K. Bera, N. Kumar, and R. Sarkar, "S-shaped versus V-shaped transfer functions for binary Manta ray foraging optimization in feature selection problem," *Neural Comput. Appl.*, vol. 33, no. 17, pp. 11027–11041, 2021.
[22] Q. Al-Tashi, H. Rais, and S. Jadid, "Feature selection method based on grey wolf optimization for coronary artery disease classification," *Adv. Intell. Syst. Comput.*, vol. 843, no. November, pp. 257–266, 2019.
[23] P. Hu, J. S. Pan, and S. C. Chu, "Improved Binary Grey Wolf Optimizer and Its application for feature selection," *Knowledge-Based Syst.*, vol. 195, p. 105746, 2020.
[24] L. Y. Yab, N. Wahid, and R. A. Hamid, *A Modified Whale Optimization Algorithm as Filter-Based Feature Selection for High Dimensional Datasets*, vol. 457 LNNS. Springer International Publishing, 2022.
[25] P. Niu, S. Niu, N. liu, and L. Chang, "The defect of the Grey Wolf optimization algorithm and its verification method," *Knowledge-Based Syst.*, vol. 171, pp. 37–43, 2019.
[26] E. Emary, H. M. Zawbaa, and A. E. Hassanien, "Binary grey wolf optimization approaches for feature selection," *Neurocomputing*, vol. 172, pp. 371–381, 2016.
[27] M. Zhong and W. Long, "Whale optimization algorithm with nonlinear control parameter," *MATEC Web Conf.*, vol. 139, pp. 1–5, 2017.
[28] M. Abdel-Basset, G. Manogaran, D. El-Shahat, and S. Mirjalili, "A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem," *Futur. Gener. Comput. Syst.*, vol. 85, no. March, pp. 129–145, 2021.
[29] F. S. Gharehchopogh and H. Gholizadeh, "A comprehensive survey: Whale Optimization Algorithm and its applications," *Swarm Evol. Comput.*, vol. 48, no. November 2018, pp. 1–24, 2019.
[30] C. L. Blake and C. J. Merz, "UCI Machine Learning Repository," 1998. [Online]. Available: https://archive.ics.uci.edu/ml/index.php. [Accessed: 28-Nov-2021].