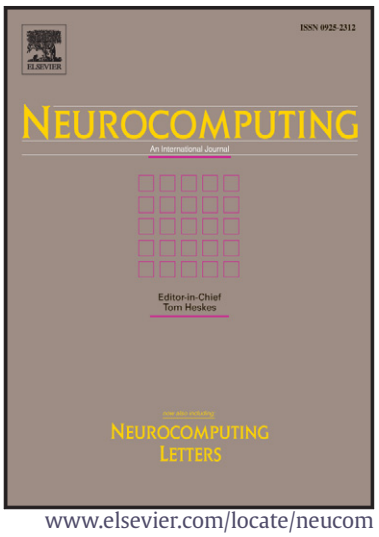


www.elsevier.com/locate/neucom

Binary Gray Wolf Optimization Approaches for Feature Selection

E. Emary, Hossam M. Zawbaa, Aboul Ella Hassanien

PII: S0925-2312(15)01050-4
DOI: http://dx.doi.org/10.1016/j.neucom.2015.06.083
Reference: NEUCOM15826

To appear in: *Neurocomputing*

Received date: 13 February 2015
Revised date: 28 June 2015
Accepted date: 29 June 2015

Cite this article as: E. Emary, Hossam M. Zawbaa, Aboul Ella Hassanien, Binary Gray Wolf Optimization Approaches for Feature Selection, *Neurocomputing*, http://dx.doi.org/10.1016/j.neucom.2015.06.083

Binary Gray Wolf Optimization Approaches for Feature Selection

E. Emary[1,*], Hossam M. Zawbaa[2,3,*], Aboul Ella Hassanien[1,2,*]

[1] *Faculty of Computers and Information, Cairo University, Egypt*
[2] *Faculty of Computers and Information, Beni-Suef University, Egypt*
[3] *Faculty of Mathematics and Computer Science, Babes-Bolyai University, Romania*
[*] *Scientific Research Group in Egypt (SRGE), http://www.egyptscience.net*

Abstract

In this work, a novel binary version of the gray wolf optimization (GWO) is proposed and used to select optimal feature subset for classification purposes. Gray wolf optimizer (GWO) is one of the latest bio-inspired optimization techniques, which simulate the hunting process of gray wolves in nature. The binary version introduced here is performed using two different approaches. In the first approach, individual steps toward the first three best solutions are binarized and then stochastic crossover is performed among the three basic moves to find the updated binary gray wolf position. In the second approach, sigmoidal function is used to squash the continuous updated position, then stochastically threshold these values to find the updated binary gray wolf position. The two approach for binary gray wolf optimization (bGWO) are hired in the feature selection domain for finding feature subset maximizing the classification accuracy while minimizing the number of selected features. The proposed binary versions were compared to two of the common optimizers used in this domain namely particle swarm optimizer and genetic algorithms. A set of assessment indicators are used to evaluate and compared the different methods over 18 different datasets from the UCI repository. Results prove the capability of the proposed binary version of gray wolf optimization (bGWO) to search the feature space for optimal feature combinations regardless of the initialization and the used stochastic operators.

Keywords: Gray wolf optimization, binary gray wolf optimization, feature selection, bio-inspired optimization, evolutionary computation.

1. Introduction

Feature selection provides a way for identifying the important features and removing irrelevant (redundant) ones from the dataset [1]. The feature selection objectives are data dimensionality reduction, improving prediction performance, and good data understanding for different machine learning applications [2]. In the real world applications, data representation often uses too many features with redundancy features, which means certain features can take the role of another and the unnecessary features can be removed. Moreover, the relevant (interdependence) features have an influence on the output and contain important information that will be obscure if any of them is excluded [3].

Previously, an exhaustive search for the optimal set of features (attributes) in a high dimensional space may be unpractical. Many researches try to model the feature selection as a combinatorial optimization problem, which the set of features lead to the best feature space separability [4]. The objective function can be the classification accuracy or some other criterion that might consider the best trade-off between attribute extraction computational burden and efficiency [5].

The classical optimization techniques have some restriction in solving the problems, so that evolutionary computation (EC) algorithms are the alternative for solving these limitations and searching for the optimum solution of the problems. Evolutionary computation (EC) algorithms are inspired from nature, social behavior, and biological behavior of (animals, birds, fish, bat, firefly, wolves, etc.) in a group. Many researchers have proposed different computational methods, in order to mimic the behavior of these species to seek for their food (optimal solution) [6].

Various heuristic techniques mimic the behaviour of biological and physical systems in the nature and it has been proposed as strong methods for global optimizations. Genetic algorithms (GA) was the first evolutionary based algorithm introduced in the literature and has been developed based on the natural process of evolution through reproduction. GA has the ability to solve the complex and non-linear problems. Moreover, GA has some disadvantages such as low performance and sticking in local minima [7]. Particle swarm optimization (PSO) is one of the well-known swarm algorithms. In particle swarm optimization (PSO), each solution is considered as a particle with specific characteristics (position, fitness, and a speed vector) which de-

fines the moving direction of the particle [8].

Artificial bee colony (ABC) is a numerical optimization algorithm based on foraging behavior of honeybees. In ABC, the employer bees try to find food source and advertise the other bees. The onlooker bees follow their interesting employer, and the scout bee fly spontaneously to find the best food source [9]. A virtual bee algorithm (VBA) is applied to optimize the numerical function in 2-D using a swarm of virtual bees, which move randomly in the search space and interact to find food sources. From the interactions between these bees results the possible solution for the optimization problem [10]. A proposed approach based on natural behavior of honeybees, which randomly generated worker bees are moved in the direction of the elite bee. The elite bee represents the optimal (near to optimal) solution [11].

In optimization algorithms, it is essential to have a convenient balance between exploration and exploitation. In a bee swarm algorithms, different behaviors of the bees give us the possibility to create robust balancing technique between exploration and exploitation [12]. Artificial fish swarm (AFS) algorithm mimics the stimulant reaction by controlling the tail and fin. AFS is a robust stochastic technique based on the fish movement and its intelligence during the food finding process [13].

A binary version of the particle swarm optimization (BPSO) modifies the old version of PSO algorithm to deal with the binary optimization problems [14]. Moreover, an extended version of BPSO is used to deal with feature selection problems [15]. The search space in BPSO is considered as a hypercube; a particle may be seen to move to nearer or farther corners of the hypercube by flipping various numbers of bits [16]. Furthermore, A binary version of the gravitational search algorithm (BGSA) is used for feature selection issue [17]. A binary version of the bat algorithm (BBA) is applied for feature selection purposes, where the search space is modelled as an n-cube. It is important to assign for every bat a set of binary coordinates that indicate if this feature will belong to the final feature set. So, the optimal (near to optimal) solution corresponds to one hypercube's corner [18]. There are many other researches with the same idea, applied in a wide range of bio-inspired algorithms [19], [20], [21], [23], [24], [25], [27], and [26].

Gray wolf optimization (GWO) is a newly introduced evolutionary algorithm, which proposes that the gray wolves have a successful reproduction

more than hunting in the pack. Two gray wolves (male and female) have a higher position and managing the other wolves in the pack [6]. In this paper, a novel binary version of the gray wolf optimization is proposed to find optimal regions of the complex search space. Gray wolf optimizer is one of the latest bio-inspired techniques, which simulate the hunting process of a pack of gray wolves in nature. The binary version introduced here is performed using two different approaches.

The organization of this paper as the following: Section 2 presents the background of continuous gray wolf optimization (CGWO). The proposed new version of gray wolf optimization (GWO) describes in section 3. Section 4 presents a binary version of gray wolf optimization (BGWO) for feature selection. The experimental results are discussed in section 5. Finally, conclusions are stated in section 6.

2. Continuous Gray Wolf Optimization (CGWO)

Mostly, grey wolves prefer to live in a pack. The group size is 5 to 12 on average. They have very strict rules in social dominant hierarchy. According to [22] gray wolf pack consists of the following:

1. The alphas are leading the pack, the alpha wolves are responsible for making decisions. The alphasdecisions are dictated to the pack
2. The betas are subordinate wolves that help the alpha in decision making or other activities. The beta can be either male or female, and he/she is probably the best candidate to be the alpha
3. The omega play the role of scapegoat. Omega wolves always have to submit to all the other dominant wolves. They are the last wolves that are allowed to eat.
4. The deltas have to submit alphas and betas, but they dominate the omega. *Scouts*, *sentinels*, *elders*, *hunters*, and *caretakers* belong to this category. *Scouts* are responsible for watching the boundaries of the territory and warning the pack in case of any danger. *Sentinels* protect and guarantee the safety of the pack. *Elders* are the experienced wolves who used to be alpha or beta. *Hunters* help the alphas and betas when hunting prey and providing food for the pack. Finally, the *caretakers* are responsible for caring for the weak, ill, and wounded wolves in the pack.

In the mathematical model for the GWO the fittest solution is called the alpha (α). The second and third best solutions are named beta (β) and delta (δ) respectively. The rest of the candidate solutions are assumed to be omega (ω). The hunting is guided by α, β, δ, and ω follow these three candidates.

In order for the pack to hunt a prey they first encircling it. In order to mathematically model encircling behavior the following equations (1), (2), (3), and (4) are used.

$$\overrightarrow{X}(t+1) = \overrightarrow{X}_p(t) + \overrightarrow{A}.\overrightarrow{D}, \tag{1}$$

where $\overrightarrow{D}$ is as defined in equation (2), t is the iteration number, $\overrightarrow{A}$ and $\overrightarrow{C}$ are coefficient vectors, $\overrightarrow{X}_p$ is the prey position, and $\overrightarrow{X}$ is the gray wolf position.

$$\overrightarrow{D} = |\overrightarrow{C}.\overrightarrow{X}_p(t) - \overrightarrow{X}(t)|, \tag{2}$$

the $\overrightarrow{A}$, $\overrightarrow{C}$ vectors are calculated as in equations (3) and (4).

$$\overrightarrow{A} = 2a.\overrightarrow{r_1} - a \tag{3}$$

$$\overrightarrow{C} = 2\overrightarrow{r_2}, \tag{4}$$

where a is linearly decreased from 2 to 0 over the course of iterations, and r_1, r_2 are random vectors in [0, 1]. The hunt is usually guided by the alpha. The beta and delta might also participate in hunting occasionally. In order to mathematically simulate the hunting behavior of grey wolves, the alpha (best candidate solution) beta (the second best candidate solution), and delta (the third best candidate solution) are assumed to have better knowledge about the potential location of prey. The first three best candidate solutions obtained so far and oblige the other search agents (including the omegas) to update their positions according to the position of the best search agents. So the updating for the wolves positions is as in equation (5).

$$\overrightarrow{X}(t+1) = \frac{\overrightarrow{X_1} + \overrightarrow{X_2} + \overrightarrow{X_3}}{3}, \tag{5}$$

where $\overrightarrow{X_1}, \overrightarrow{X_2}, \overrightarrow{X_3}$ are defined as in equations (6), (7), and (8) respectively.

$$\overrightarrow{X_1} = |\overrightarrow{X_\alpha} - \overrightarrow{A_1}.\overrightarrow{D_\alpha}|, \tag{6}$$

$$\overrightarrow{X_2} = |\overrightarrow{X_\beta} - \overrightarrow{A_2}.\overrightarrow{D_\beta}|, \tag{7}$$

$$\overrightarrow{X_3} = |\overrightarrow{X_\delta} - \overrightarrow{A_3}.\overrightarrow{D_\delta}|, \tag{8}$$

where $\overrightarrow{X_\alpha}, \overrightarrow{X_\beta}, \overrightarrow{X_\delta}$ are the first three best solutions in the swarm at a given iteration t, $\overrightarrow{A_1}, \overrightarrow{A_2}, \overrightarrow{A_3}$ are defined as in equation (3), and $\overrightarrow{D_\alpha}, \overrightarrow{D_\beta}, \overrightarrow{D_\gamma}$ are defined using equations (9), (10), and (11) respectively.

$$\overrightarrow{D_\alpha} = |\overrightarrow{C_1}.\overrightarrow{X_\alpha} - \overrightarrow{X}|, \tag{9}$$

$$\overrightarrow{D_\beta} = |\overrightarrow{C_2}.\overrightarrow{X_\beta} - \overrightarrow{X}|, \tag{10}$$

$$\overrightarrow{D_\delta} = |\overrightarrow{C_3}.\overrightarrow{X_\delta} - \overrightarrow{X}|, \tag{11}$$

where $\overrightarrow{C_1}, \overrightarrow{C_2}, and \overrightarrow{C_3}$ are defined as in equation (4).

A final remark about the gray wolf optimizer (GWO) is the updating of the parameter a that controls the tradeoff between *exploration* and *exploitation*. The parameter a is linearly updated in each iteration to range from 2 to 0 according to the equation (12).

$$a = 2 - t\frac{2}{MaxIter}, \tag{12}$$

where t is the iteration number and $MaxIter$ is the total number of iteration allowed for the optimization. Algorithm 1 outlines the continuous gray wolf optimization (CGWO) algorithm.

3. The Proposed Binary Gray Wolf Optimization

In the continuous gray wolf optimization (CGWO) wolves continuously change their positions to whatever point in the space. In some special problems such as feature selection the solutions are restricted to the binary $\{0, 1\}$ values which motivates a special version of the CGWO. In this work, a novel binary gray wolf optimization (bGWO) is proposed for the feature selection task. The wolves updating equation is a function of three position vectors

input : n Number of gray wolves in the pack,
N_{Iter} Number of iterations for optimization.
output: x_α Optimal gray wolf position,
$f(x_\alpha)$ Best fitness value.

1. Initialize a population of n gray wolves positions randomly.
2. Find the α, β, and δ solutions based on their fitness values.
3. **while** *Stopping criteria not met* **do**
   foreach $Wolf_i \in pack$ **do**
   Update current wolf's position according to equation (5).
   end
   I Update a, A, and C.
   II Evaluate the positions of individual wolves.
   III Update α, β, and δ.
   end

Algorithm 1: Continuous gray wolf optimization algorithm

namely $x_\alpha, x_\beta, x_\delta$ which attracts each wolf towards the first three best solutions. In the bGWO, the pool of solutions is in binary form at any given time; all solutions are on the corner of a hypercube. To update the positions of a given wolf according to the CGWO principle, while keeping the binary restriction based on equation (5). We can apply one of two approaches, which describe in more details in the following subsections.

3.1. Binary Gray Wolf Optimization - Approach 1 (bGWO1)

In this approach bGWO1 the main updating equation can be formulated as shown in equation (13); see algorithm 2.

$$X_i^{t+1} = Crossover(x_1, x_2, x_3), \tag{13}$$

where $Crossover(x, y, z)$ is suitable cross over between solutions x, y, z and x_1, x_2, x_3 are binary vectors representing the effect of wolf move towards the $alpha, beta, delta$ gray wolves in order. x_1, x_2, x_3 are calculated using equations (14), (17), and (20) respectively.

$$x_1^d = \begin{cases} 1 & if(x_\alpha^d + bstep_\alpha^d) \geq 1 \\ 0 & otherwise \end{cases} \tag{14}$$

where x_α^d is the position vector of the alpha wolf in the dimension d, and $bstep_\alpha^d$ is a binary step in dimension d that can be calculated as in equation (15).

$$bstep_\alpha^d = \begin{cases} 1 & \text{if } cstep_\alpha^d \geq rand \\ 0 & otherwise \end{cases} \tag{15}$$

where $rand$ is a random number drawn from uniform distribution $\in [0, 1]$, and $cstep_\alpha^d$ is the continuous valued step size for dimension d and can be calculated using sigmoidal function as in equation 16.

$$cstep_\alpha^d = \frac{1}{1 + e^{-10(A_1^d D_\alpha^d - 0.5)}}, \tag{16}$$

where A_1^d, D_α^d are calculated using equations (3), and (9) in the dimension d.

$$x_2^d = \begin{cases} 1 & if(x_\beta^d + bstep_\beta^d) \geq 1 \\ 0 & otherwise \end{cases} \tag{17}$$

where x_β^d is the position vector of the beta wolf in the dimension d, and $bstep_\beta^d$ is a binary step in dimension d that can be calculated as in equation (18).

$$bstep_\beta^d = \begin{cases} 1 & \text{if } cstep_\beta^d \geq rand \\ 0 & otherwise \end{cases} \tag{18}$$

where $rand$ is a random number drawn from uniform distribution $\in [0, 1]$, and $cstep_\beta^d$ is the continuous valued step size for dimension d and can be calculated using sigmoidal function as in equation (19).

$$cstep_\beta^d = \frac{1}{1 + e^{-10(A_1^d D_\beta^d - 0.5)}}, \tag{19}$$

where A_1^d, and D_β^d are calculated using equations (3), (9) in the dimension d.

$$x_3^d = \begin{cases} 1 & if(x_\delta^d + bstep_\delta^d) \geq 1 \\ 0 & otherwise \end{cases} \tag{20}$$

where x_δ^d is the position vector of the delta wolf in the dimension, d and $bstep_\delta^d$ is a binary step in dimension d that can be calculated as in equation (21).

$$bstep_\delta^d = \begin{cases} 1 & \text{if } cstep_\delta^d \geq rand \\ 0 & otherwise \end{cases} \tag{21}$$

where $rand$ is a random number drawn from uniform distribution $\in [0, 1]$, and $cstep_\delta^d$ is the continuous valued step size for dimension d and can be calculated using sigmoidal function as in equation (22).

$$cstep_\delta^d = \frac{1}{1 + e^{-10(A_1^d D_\delta^d - 0.5)}}, \tag{22}$$

where A_1^d, D_δ^d are calculated using equations 3, 9 in the dimension d.

A simple stochastic crossover strategy is applied per dimension to crossover a, b, c solutions as shown in equation (23).

$$x_d = \begin{cases} a_d & if rand < \frac{1}{3} \\ b_d & \frac{1}{3} \leq rand < \frac{2}{3} \\ c_d & otherwise \end{cases} \tag{23}$$

Where a_d, b_d, c_d are the binary values for first, second and third parameter in dimension d, x_d is the crossover output at dimension d, and $rand$ is a random number drawn from uniform distribution in the range [0, 1].

3.2. Binary Gray Wolf Optimization - Approach 2 (bGWO2)

In this approach only the updated gray wolf position vector is forced to be binary; see algorithm 3 using the main updating equation as shown in equation (24).

$$x_d^{t+1} = \begin{cases} 1 & \text{if } sigmoid(\frac{x_1 + x_2 + x_3}{3}) \geq rand \\ 0 & otherwise \end{cases} \tag{24}$$

input : n Number of gray wolves in the pack,
N_{Iter} Number of iterations for optimization.
output: x_α Optimal gray wolf binary position,
$f(x_\alpha)$ Best fitness value.

1. Initialize a population of n wolves positions at random $\in [0, 1]$.
2. Find the α, β, δ solutions based on fitness.
3. **while** *Stopping criteria not met* **do**

foreach $Wolf_i \in pack$ **do**
Calculate x_1, x_2, x_3 using equations (14), (17), and (20)
$x_i^{t+1} \leftarrow$ crossover among x_1, x_2, x_3 using equation (23).
end

I Update a, A, C.

II Evaluate The positions of individual wolves.

III Update α, β, δ.

end

Algorithm 2: Binary Gray wolf optimization algorithm - Approach 1 (bGWO1)

where *rand* is a random number drawn from uniform distribution $\in [0, 1]$, x_d^{t+1} is the updated binary position in dimension d at iteration t, and $sigmoid(a)$ is defined in equation (25).

$$sigmoid(a) = \frac{1}{1 + e^{-10(x-0.5)}} \tag{25}$$

input : n Number of gray wolves in the pack,
N_{Iter} Number of iterations for optimization.
output: x_α Optimal gray wolf binary position,
$f(x_\alpha)$ Best fitness value.

1. Initialize a population of n wolves positions at random $\in [0, 1]$.
2. Find the α, β, δ solutions based on fitness.
3. **while** *Stopping criteria not met* **do**

foreach $Wolf_i \in pack$ **do**
Update $wolf_i$ position to a binary position according to equation (24).
end

I Update a, A, C.

II Evaluate The positions of individual wolves.

III Update α, β, δ.

end

Algorithm 3: Binary Gray Wolf Optimization Algorithm - Approach 2 (bGWO2)

4. Binary Gray Wolf Optimization for Feature Selection

In this section the two binary version s of gray wolf optimization (bGWO1 and bGWO2) are exploited in feature selection for classification problems. For a feature vector sized N the different feature reducts would be 2^N which is a huge space of features to be searched *exhaustively*. So, the binary gray wolf optimization is used to adaptively search the feature space for best feature combination. The best feature combination is the one with maximum *classification performance* and minimum *number of selected features*. The

fitness function used in binary gray wolf optimization to evaluate individual gray wolf positions is as shown in equation (26).

$$Fitness = \alpha\gamma_R(D) + \beta\frac{|\ C - R\ |}{|\ C\ |}, \tag{26}$$

where $\gamma_R(D)$ is the classification quality of condition attribute set R relative to decision D, R is the length of selected feature subset, C is the total number of features, α and β are two parameters corresponding to the importance of classification quality and subset length, $\alpha\in[0,1]$ and $\beta = 1 - \alpha$. We can see that the fitness function maximizes the classification quality; $\gamma_R(D)$, and the ratio of the unselected features to the total number of features; $\frac{|C-R|}{|C|}$.

The above equation can be easily converted into a minimization problem by using error rate rather than classification quality and using selected features ration rather than than using unselected feature size. The minimization problem can be formulated as in equation (27).

$$Fitness = \alpha E_R(D) + \beta\frac{|\ R\ |}{|\ C\ |}, \tag{27}$$

where $E_R(D)$ is the error rate for the classifier of condition attribute set, R is the length of selected feature subset, and C is the total number of features. $\alpha\in[0,1]$ and $\beta = 1 - \alpha$ are constants to control the importance of classification accuracy and feature reduction; $\beta = 0.01$ in current experiments.

The main characteristic of wrapper methodologies in feature selection is the use of the classifier as guide of feature selection procedure. Wrapper based feature selection can be classified based on the following three main items:

1. Classification method.
2. Feature evaluation criteria.
3. Search method.

K-nearest neighbor (KNN) [28] is a common simple method used for classification. KNN is a supervised learning algorithm that classifies an unknown sample instance based on the majority of the K-nearest neighbor category. Classifiers do not use any model for K-nearest neighbors and are determined

solely based on the minimum distance from the query instance to the training samples. The KNN method is simple and easy to implement method and hence it is very common classifier. In this proposed system, the KNN is used as a classification to ensure the goodness of the selected features.

Binary gray wolf optimization is used as a search method that can adaptively search the feature space for maximizing the feature evaluation criteria; as shown in equation (26). A single dimension in the search space represents individual feature and hence the wolfś position represents a single feature combination or solution.

5. Experimental results and discussions

5.1. Data Description

Eighteen datasets in table (1) from the UCI machine learning repository [29] are used in the experiments and comparisons results. The datasets were selected to have various numbers of attributes and instances as representatives of various kinds of issues that the proposed technique will be tested on. For each dataset, the instances are randomly divided into three sets namely training, validation and testing sets in cross validation manner.

A wrapper approach for feature selection is used in this study based on KNN classifier. A simple and commonly utilized learning algorithm [28], KNN is utilized in the experiments based on trial and error basis where the best choice of K is selected ($K = 5$) as the best performing on all the datasets. Through the training process, every wolf position represents one attribute subset. Training set is used to evaluate the KNN on the validation set throughout the optimization to guide the feature selection process. The test data are kept hidden from the optimization and is let for final evaluation. The proposed feature selection methods are benchmarked with particle swarm optimization (PSO) [30] and genetic algorithm (GA) [31] for evaluation. The global and optimizer-specific parameter setting is outlined in table (2). All the parameters are set either according to domain specific-knowledge as the α, β parameters of the used fitness function, or based on trial and error on small simulations and common in literature such as the rest of parameters.

5.2. Evaluation Criteria

Individual datasets are divided randomly into 3 different equal portions namely validation, training, and testing datasets. The partitioning of the data is repeated for 20 times to ensure stability and statistical significance

Table 1: Datasets description

Dataset	No. Attributes	No. Instances
Breastcancer	9	699
Exactly	13	1000
Exactly2	13	1000
Lymphography	18	148
M-of-n	13	1000
Tic-tac-toe	9	958
Vote	16	300
Zoo	16	101
WineEW	13	178
SpectEW	22	267
SonarEW	60	208
PenglungEW	325	73
IonosphereEW	34	351
HeartEW	13	270
Congress	16	435
BreastEW	30	569
KrvskpEW	36	3196
WaveformEW	40	5000

of the results. In each run, the following measures are recorded from the validation data:

- *Classification average accuracy* is an indicator describes how accurate is the classifier given the selected feature set. The classification average accuracy can be formulated in equation (28).

$$AvgPerf = \frac{1}{M}\sum_{j=1}^{M}\frac{1}{N}\sum_{i=1}^{N} Match(C_i, L_i), \tag{28}$$

where M is the number of times to run the optimization algorithm to select feature subset, N is the number of points in the test set, C_i is the classifier output label for data point i, L_i is the reference class label

Table 2: Parameter setting for experiments

Parameter	Value
No of search agents	8
No of iterations	70
Problem dimension	Number of features in the data
Search domain	[0 1]
No. Repetitions of runs	20
Crossover Fraction in GA	0.8
inertia factor of PSO	0.1
individual-best acceleration factor of PSO	0.1
α parameter in the fitness function	0.99
β parameter in the fitness function	0.01

for data point i, and $Match$ is a function that outputs 1 when the two input labels are the same and outputs 0 when they are different.

- *Statistical best* is the minimum fitness function obtained for a given optimizer at the different M operations of an optimization algorithm. Best represents the most optimistic solution acquired and can be formulated in equation (29).

$$best = min_{i=1}^{M} g_*^i, \tag{29}$$

where M is the number of times to run the optimization algorithm to select feature subset, and g_*^i is the optimal solution resulted from run number i.

- *Statistical worst* is the worst solution among the best solutions found for running an optimization algorithm for M times. Worst represents the pessimistic solution and can be formulated in equation (30).

$$Worst = min_{i=1}^{M} g_*^i, \tag{30}$$

where M is the number of times to run the optimization algorithm to select feature subset, and g_*^i is the optimal solution resulted from run

number i.

- *Statistical mean* is the average of solutions acquired from running an optimization algorithm for different M running. Mean represents the average performance a given stochastic optimizer can be formulated in equation (31).

$$Mean = \frac{1}{M}\sum_{i=1}^{M} g_*^i, \tag{31}$$

where M is the number of times to run the optimization algorithm to select feature subset, and g_*^i is the optimal solution resulted from run number i.

- *Std* is a representation for the variation of the obtained best solutions found for running a stochastic optimizer for M different runs. *Std* is used as an indicator for optimizer stability and robustness, whereas *Std* is smaller this means that the optimizer converges always to same solution; while larger values for *Std* mean much random results. *Std* is formulated as in equation (32).

$$Std = \sqrt{\frac{1}{M-1}\sum (g_*^i - Mean)^2}, \tag{32}$$

where M is the number of times to run the optimization algorithm to select feature subset, g_*^i is the optimal solution resulted from run number i, and $Mean$ is the average defined in equation (31).

- *average selection size* represents the average size of the selected features to the total number of features. This measure can be formulated as in equation (33).

$$AVGSelectionSZ = \frac{1}{M}\sum_{i=1}^{M} \frac{size(g_*^i)}{D}, \tag{33}$$

where M is the number of times to run the optimization algorithm to select feature subset, g_*^i is the optimal solution resulted from run number i, $size(x)$ is the number of on values for the vector x, and D is the number of features in the original dataset.

- *Average F-Score* is a measure that evaluate a feature subset such that in the data space spanned by the selected features, the distances between data points in different classes are as large as possible, while the distances between data points in the same class are as small as possible [32]. The Fisher score in this work is calculated for individual features given the class labels as in equation (34).

$$F_j = \frac{\sum_{k=1}^{c} n_k(\mu_k^j - \mu^j)^2}{(\sigma^j)^2}, \tag{34}$$

  where F_j is the fisher index for feature j, $\mu^j, (\sigma^j)^2$ is the mean and standard deviation of the whole data set, n_k is the size of class k, and μ_k^j is the mean of class k. The average F-score is calculated as the average of score sum of optimal solution found by running individual optimizers for M times.

- *Wilcoxon rank sum test* proposed by Frank Wilcoxon [33] as a nonparametric test. The test assigns ranks to all the scores considered as one group, and then sums the ranks of each group. The null hypothesis is that the two samples come from the same population, so any difference in the two rank sums comes only from sampling error. The rank-sum test is often described as the nonparametric version of the t test for two independent groups. It tests the null hypothesis that data in x and y vectors are samples from continuous distributions with equal medians, against the alternative that they are not.

The proposed two binary versions of gray wolf optimizers (bGWO1 and bGWO2) are compared against two of the very common optimizers namely particle swarm optimization (PSO) and genetic algorithms (GA), and three different methods are used to initialize the different optimization algorithms to ensure convergence namely *random initialization*, *small initialization*, and *large initialization*.

1. *Small initialization*: search agents are initialized with the minor number of random selected features. Therefore, if the number of agents is less than number of features we will see that each search agent will have a single dimension with 1. Of course, the optimizer will search for feature(s) to be set to 1 to enhance the fitness function value as in the standard forward selection of features; as shown in figure (1-a).

2. *Random initialization*: where all wolves' positions are randomly initialized with a 0/1, values at each dimension, as shown in figure (1-b).
3. *Large initialization*: search agents are set to its maximum. In this case, we find a solution with all dimensions set to 1 and solutions with all dimensions set to 1 except for single random dimension set to 0. Of course, the optimizer will search for feature(s) to be removed while keeping or enhancing the fitness function value, as shown in figure (1-c).

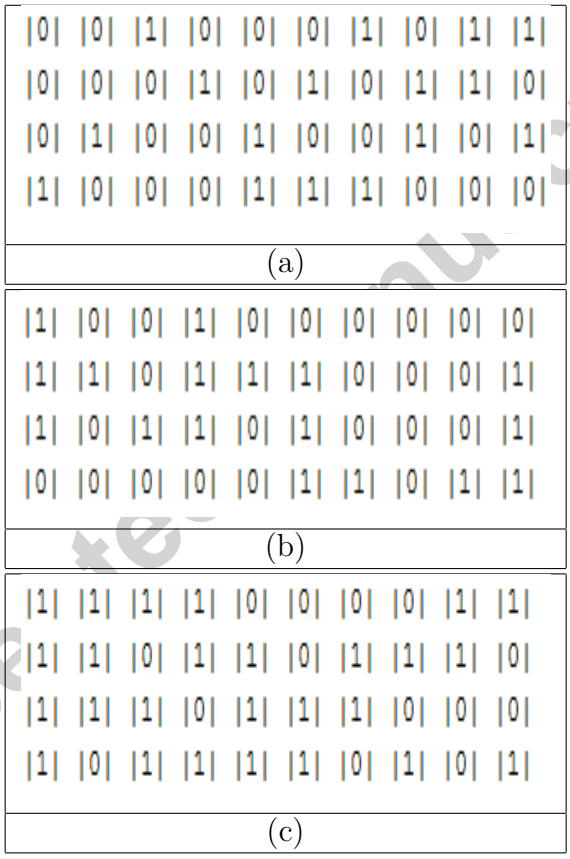


Figure 1: Sample initial gray wolves positions using small, normal and large initialization

5.3. Performance on Uniform Random Initialization

Table (3) outlines the performance of the different methods namely genetic algorithms (GA), particle swarm optimization (PSO), binary gray wolf

optimization - approach 1 (bGWO1), and binary gray wolf optimization - approach 2 (bGWO2) using *random initialization* using fitness function defined in equation (27) in a *minimization* mode. The table outlines the average fitness obtained over the different runs.

We can remark that the best performance is achieved by the proposed bGWO2 in the obtained fitness value, which proves the capability of the bGWO2 for searching the feature space adaptively better than the other methods. The similar results can be seen in tables (4) and (5) which outline the best and worst obtained fitness function over all the runs.

Table 3: Mean fitness function obtained from the different optimizers using uniform initialization

Dataset	bGWO1	bGWO2	GA	PSO
Breastcancer	0.030	**0.027**	0.027	0.028
BreastEW	0.037	0.031	**0.027**	0.033
CongressEW	0.057	**0.037**	0.044	0.048
Exactly	0.315	**0.217**	0.291	0.277
Exactly2	0.244	0.246	**0.242**	0.249
HeartEW	0.136	**0.127**	0.142	0.144
IonosphereEW	0.106	**0.084**	0.111	0.101
KrvskpEW	0.065	**0.038**	0.047	0.055
Lymphography	0.196	**0.151**	0.168	0.159
M-of-n	0.135	**0.038**	0.067	0.068
PenglungEW	0.242	**0.225**	0.250	0.250
SonarEW	0.174	**0.104**	0.154	0.154
SpectEW	0.178	**0.153**	0.160	0.166
Tic-tac-toe	0.233	0.225	0.233	**0.223**
Vote	0.056	**0.028**	0.040	0.042
WaveformEW	0.214	**0.200**	0.206	0.221
WineEW	0.044	**0.014**	0.020	0.034
Zoo	0.127	**0.112**	0.118	0.133
Total	2.587	**2.055**	2.347	2.384

The performance of the proposed bGWO1 is asserted also on the *test* data as shown in table (6), where the table outlines the classification accuracy on

Table 4: Best fitness function obtained from the different optimizers using uniform initialization

DataSet	bGWO1	bGWO2	GA	PSO
Breastcancer	**0.017**	**0.017**	**0.017**	**0.017**
BreastEW	0.021	0.016	**0.005**	0.016
CongressEW	0.041	**0.021**	0.028	0.034
Exactly	0.275	**0.069**	0.257	0.180
Exactly2	**0.213**	0.234	0.234	0.234
HeartEW	0.122	**0.100**	0.111	0.122
IonosphereEW	0.085	**0.068**	0.094	0.085
KrvskpEW	0.056	**0.022**	0.035	0.031
Lymphography	0.143	**0.102**	0.122	0.143
M-of-n	0.087	**0.000**	0.021	**0.000**
PenglungEW	0.167	0.167	0.167	**0.125**
SonarEW	0.145	**0.072**	**0.072**	0.101
SpectEW	0.146	**0.124**	**0.124**	0.146
Tic-tac-toe	0.216	0.206	**0.200**	0.212
Vote	0.030	**0.000**	**0.000**	**0.000**
WaveformEW	0.206	**0.186**	0.199	0.205
WineEW	**0.000**	**0.000**	**0.000**	**0.000**
Zoo	0.077	**0.000**	**0.000**	0.077
Total	2.047	**1.404**	1.686	1.729

test data; on the average, based on the selected features by the different optimizers. We can remark that the performance of the bGWO2 overcomes the obtained results for bGWO1, PSO, and GA, which proves its future performance on the unseen data, and hence it can be used as a candidate for feature selection.

To assess data separability and compactness based on the selected features $fisher$ index is calculated on the average for selected features from the different optimizers; as shown in table (7). We can see from the table that the best optimizer achieving data compactness is the bGWO1, which conforms that it can perform better for classification tasks.

Table (8) outline the mean selected features ratio for the different opti-

Table 5: Worst fitness function obtained from the different optimizers using uniform initialization

DataSet	bGWO1	bGWO2	GA	PSO
Breastcancer	0.043	0.039	0.039	**0.034**
BreastEW	0.053	**0.042**	0.047	0.047
CongressEW	0.083	**0.062**	0.069	0.076
Exactly	0.335	**0.317**	0.326	0.323
Exactly2	0.260	0.263	**0.251**	0.260
HeartEW	**0.156**	**0.156**	0.167	0.189
IonosphereEW	0.120	**0.103**	0.128	0.120
KrvskpEW	0.078	**0.059**	0.064	0.072
Lymphography	0.265	0.204	0.204	**0.184**
M-of-n	0.201	0.144	0.141	**0.123**
PenglungEW	0.417	**0.375**	0.458	0.417
SonarEW	0.246	**0.145**	0.246	0.203
SpectEW	0.213	0.202	0.202	**0.180**
Tic-tac-toe	0.256	0.241	0.253	**0.231**
Vote	0.070	**0.050**	0.060	0.080
WaveformEW	0.232	**0.208**	0.221	0.229
WineEW	0.119	**0.051**	**0.051**	**0.051**
Zoo	**0.176**	**0.176**	**0.176**	**0.176**
Total	3.323	**2.837**	3.105	2.995

mizers. We can see that the performance of bGWO2 is superior in selecting less number of features while keeping its good classification performance. This proves the capability of the bGWO1 for searching capability for both the objectives of the optimization and can be considered as a candidate for selecting minimum number of features achieving superior performance.

For testing the robustness and repeatability of convergence of the stochastic algorithms, the standard deviation of the obtained fitness values is calculated on the 20 runs and outlined in table (9). We can see that the standard deviation is minimum for the bGWO2 and PSO, which proves their stability, repeatability, and ability to reach optimal regardless of the used randomness and initial positions of the search agents.

Table 6: Average performance of the features selected by the different optimizers on the test data using uniform initialization

Dataset	bGWO1	bGWO2	genetic	PSO
Breastcancer	**0.976**	0.975	0.968	0.967
BreastEW	0.924	0.935	**0.939**	0.933
CongressEW	0.935	**0.938**	0.932	0.928
Exactly	0.708	**0.776**	0.674	0.688
Exactly2	0.745	**0.750**	0.746	0.730
HeartEW	0.776	0.776	0.780	**0.787**
IonosphereEW	0.807	**0.834**	0.814	0.819
KrvskpEW	0.944	**0.956**	0.920	0.941
Lymphography	**0.744**	0.700	0.696	**0.744**
M-of-n	0.908	**0.963**	0.861	0.921
PenglungEW	**0.600**	0.584	0.584	0.584
SonarEW	0.731	0.729	**0.754**	0.737
SpectEW	0.820	**0.822**	0.793	**0.822**
Tic-tac-toe	0.728	0.727	0.719	**0.735**
Vote	0.912	**0.920**	0.904	0.904
WaveformEW	0.786	**0.789**	0.773	0.762
WineEW	0.930	0.920	**0.937**	0.933
Woo	**0.879**	**0.879**	0.855	0.861
Total	14.853	**14.973**	14.649	14.796

5.4. Performance on Large and Small Initialization

Figure (2) outlines the best, mean and worst acquired fitness function value averaged over all the data sets using the small and large initialization. We can see from the figure that the bGWO2 is still performing better than PSO and GA, which confirms the searching capability of bGWO2. We can also remark that the large initialization is performing better than small initialization regardless of the used optimizer. This can be interpreted by that the large initialization initializes the optimizer with solutions close to the optimal that is commonly most of the features selected.

Figure (3) outlines the performance on test data averaged over all the data sets using the selected features from the different optimizers. We can see from the figure that the bGWO2 still performing better than other method. We

Table 7: Average Fisher Index of the features selected by the different optimizers on the test data using uniform initialization

Dataset	bGWO1	bGWO2	GA	PSO
Breastcancer	**7.033**	6.513	6.266	6.241
BreastEW	**10.047**	6.949	9.196	8.323
CongressEW	3.541	3.182	3.499	**4.529**
Exactly	**0.021**	0.014	0.021	0.019
Exactly2	**0.022**	0.012	0.011	0.015
HeartEW	1.359	1.083	1.279	**1.372**
IonosphereEW	**1.720**	1.002	1.201	1.287
KrvskpEW	**0.851**	0.695	0.749	0.699
Lymphography	**4.562**	4.247	2.413	2.884
M-of-n	**0.394**	0.382	0.391	0.389
PenglungEW	119.911	95.454	119.295	**127.077**
SonarEW	**1.757**	0.917	1.497	1.328
SpectEW	**0.719**	0.519	0.645	0.661
Tic-tac-toe	**0.057**	0.045	0.047	0.046
Vote	**3.631**	2.453	3.512	3.442
WaveformEW	**5.832**	4.891	5.597	4.786
WineEW	**8.549**	7.380	7.960	7.285
Zoo	228.514	180.138	210.540	**239.934**

can also remark that regardless of the used initialization method the bGWO2 performs better than PSO and GA. We can also remark that regardless of the used optimizer the initialization using the large initialization is better than the small initialization one.

Regarding the size of the selected features, figure (4) outlines the average feature reduction over all the data sets for the different optimizers. We can remark that the bGWO1 obtains the best reduction regardless of the used initialization method. We can also remark that as expected the large initialization outputs less reduction of features.

To assess the repeatability and robustness, figure (5) outlines the std measure averaged over the different used data sets using different optimizers. We can see that the std measure is comparable for the two used initialization

Table 8: Average selected feature ratio on the average for the different optimizers using uniform initialization

Dataset	bGWO1	bGWO2	GA	PSO
Breastcancer	0.644	**0.511**	0.556	0.556
BreastEW	0.700	**0.433**	0.600	0.580
CongressEW	0.438	**0.412**	**0.412**	0.563
Exactly	0.662	**0.400**	0.662	0.600
Exactly2	0.646	**0.338**	0.400	0.462
HeartEW	0.708	**0.477**	0.662	0.677
IonosphereEW	0.576	**0.271**	0.482	0.506
KrvskpEW	0.739	**0.356**	0.528	0.472
Lymphography	0.533	0.489	**0.456**	0.544
M-of-n	0.815	**0.462**	0.600	0.523
PenglungEW	0.494	**0.383**	0.489	0.513
SonarEW	0.620	**0.267**	0.517	0.510
SpectEW	0.582	**0.391**	0.482	0.509
Tic-tac-toe	0.800	**0.578**	**0.578**	0.644
Vote	0.537	**0.300**	0.475	0.463
WaveformEW	0.750	**0.365**	0.540	0.570
WineEW	0.677	**0.508**	0.554	0.554
Zoo	0.662	**0.463**	0.600	0.575
Average	0.644	**0.411**	0.533	0.546

method and that the bGWO2 has comparable std measure with other used optimizers.

Table (10) outlines the *Wilcoxon* test calculated on the average fitness obtained by the different optimizers on all the data sets. So, individual elements presented to the *Wilcoxon* are calculated using the formula:

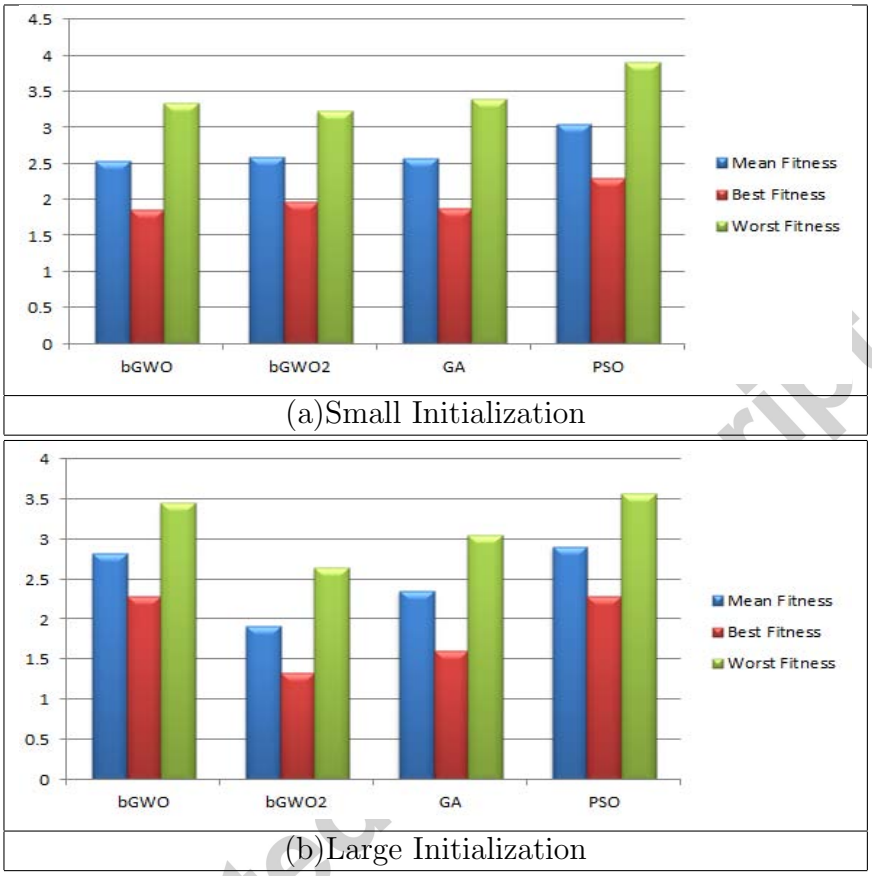
$$W_{jk} = \frac{1}{M} \sum_{i=1}^{M} Fitness_{ijk} \tag{35}$$

where M is the number of data sets used, j is the optimizer used for evaluation; bGWO-bGWO2-PSO-GA, k is the run number; ranges from 1 to 20, and W_{jk} is the is the individual element presented to the Wilcoxon test and

Table 9: Standard deviation of the obtained fitness function values over the 20 runs for the different methods using uniform initialization

Dataset	bGWO1	bGWO2	GA	PSO
Breastcancer	0.012	0.010	0.009	**0.007**
BreastEW	0.013	**0.011**	0.016	0.013
CongressEW	0.017	**0.016**	0.017	0.020
Exactly	**0.024**	0.113	0.025	0.059
Exactly2	0.021	0.013	**0.008**	0.011
HeartEW	**0.014**	0.028	0.021	0.026
IonosphereEW	0.016	0.014	**0.012**	0.014
KrvskpEW	**0.010**	0.017	0.012	0.016
Lymphography	0.055	0.037	0.033	**0.017**
M-of-n	**0.041**	0.060	0.048	0.053
PenglungEW	0.104	**0.091**	0.121	0.114
SonarEW	0.042	**0.031**	0.069	0.051
SpectEW	0.027	0.032	0.029	**0.015**
Tic-tac-toe	0.015	0.012	0.021	**0.009**
Vote	0.019	**0.019**	0.025	0.029
WaveformEW	0.011	0.009	**0.009**	0.009
WineEW	0.046	0.022	0.022	**0.021**
Zoo	0.043	0.067	0.069	**0.038**

stands for performance of optimizer j on all the data sets at run k.
We can see from the results that the bGWO2 achieves significant enhance over the PSO regardless of the used initialization method achieving p-value of $0.008, 0.008, 0.023$ on the *large*, *small* and *normal* initialization indicating that test rejects the null hypothesis of equal medians at the default 5% significance level. This proves the capability of bGWO2 to find the optimal solution regardless of the initial solution and the stochastic decisions employed throughout the search.
We can also see that the performance of bGWO2 achieves significant advance over the GA in case of using *large* initialization; p-value 0.008. This can be interpreted by the capability of bGWO2 to escape from the local minima; always initial solutions at large initialization are near optimal, and find a global optima while GA can't but in the other initialization methods the



(a)Small Initialization

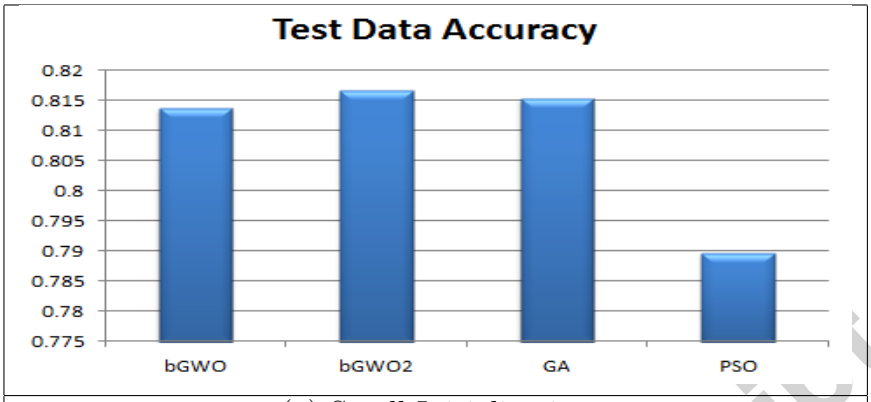
(b)Large Initialization

Figure 2: Mean, best and worst fitness obtained averaged over all the data sets from the different optimizer using the small and large initialization method

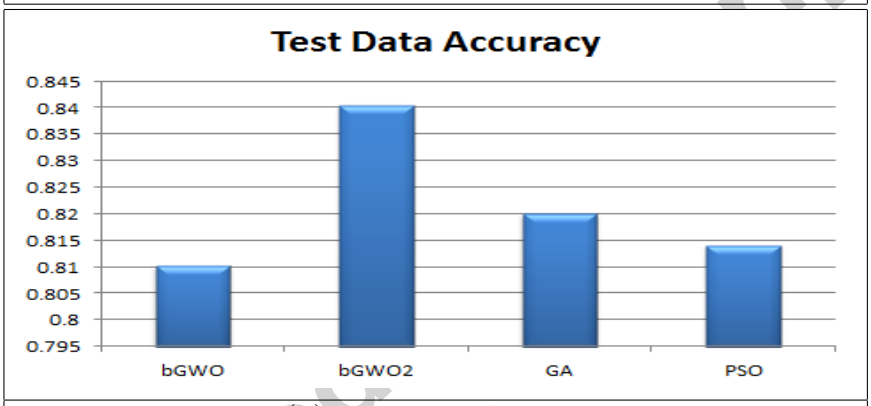
task of finding the global optima is much easier and hence global optima can be found by both GA and bGWO2.
Assessing the performance of bGWO, we can see that its significance in performance is smaller than the bGWO2 where it only achieves significant performance over PSO in case of *small* initialization; p-value 0.008, and achieves significant advance over GA in case of using *large* initialization.

6. Conclusion and Future work

In this work two novel binary versions of the gray wolf optimization method were proposed for feature selection in wrapper mode. The continu-



(a) Small Initialization

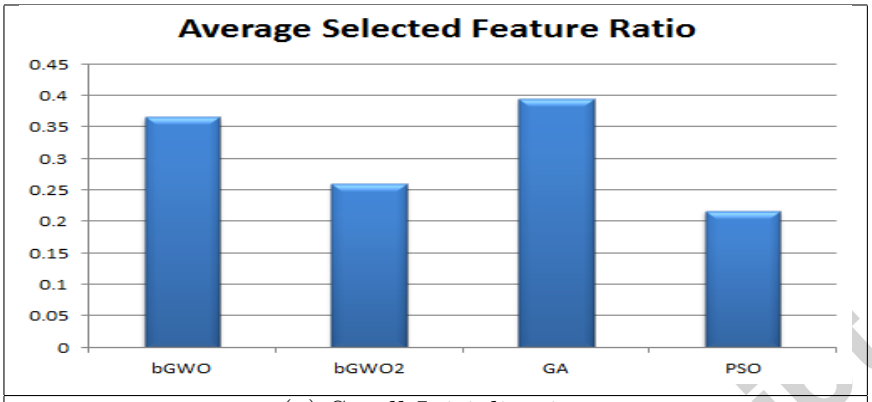


(b)Large Initialization

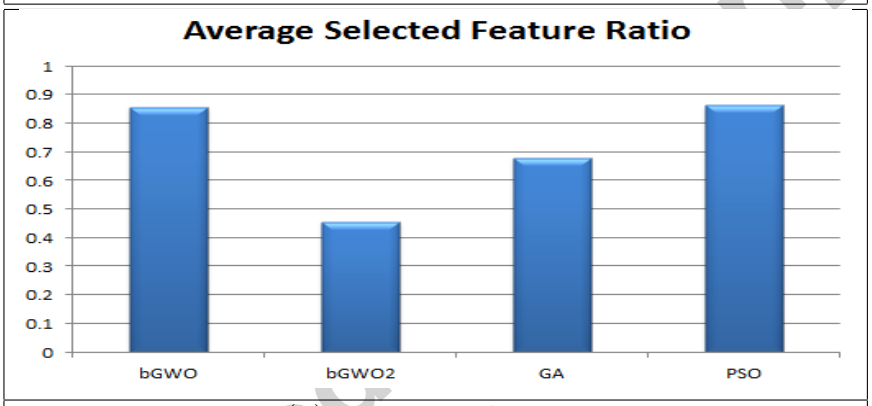
Figure 3: Average performance over the test data averaged over all the data sets using the different optimizers

ous version of gray wolf optimization (CGWO) is converted into the binary form using two approaches. The proposed two binary approaches are applied and used for feature selection in machine learning domain using different initialization methods. The two approach for binary gray wolf optimization (bGWO1 and bGWO2) are hired in the feature selection domain for evaluation and results are compared against two of the well-known feature selection methods particle swarm optimization (PSO) and genetic algorithms (GA).

The evaluation is performed using a set of evaluation criteria to assess different aspects of the proposed system. The obtained results finds out that the proposed binary version outperforms other methods in the search



(a) Small Initialization



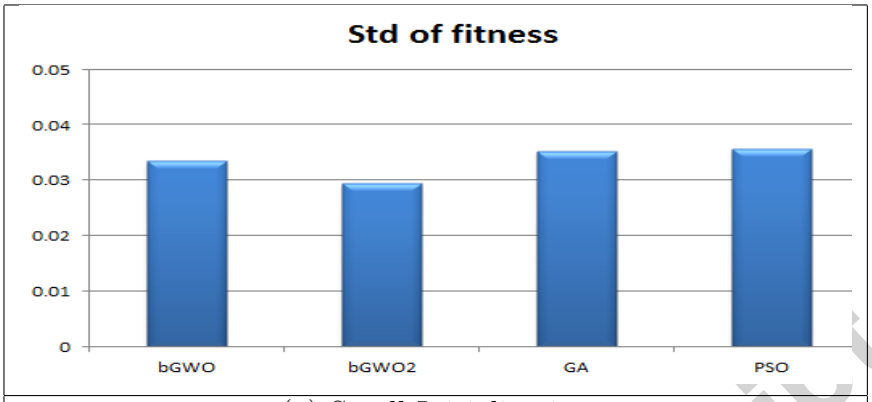
(b)Large Initialization

Figure 4: Selected feature ration averaged over all the data sets using different optimizers using both small and large initialization

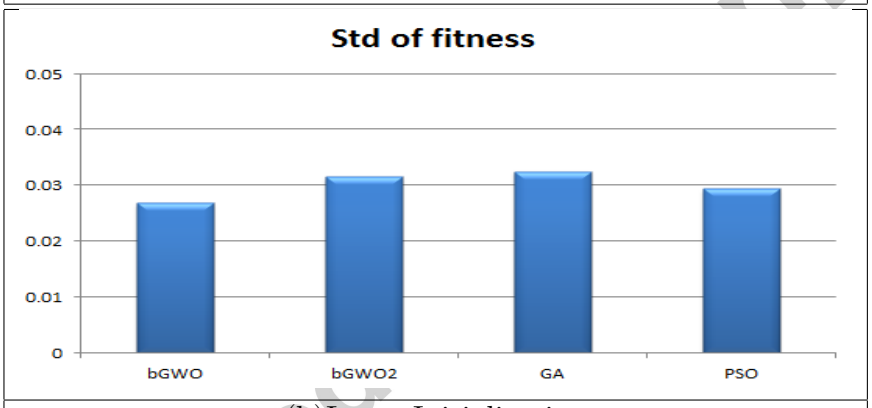
capability. We found also that the performance of the selected features on test data is best for the features selected by the proposed algorithm. To assess the data separability based on the selected features we used fisher index as an indicator and the proposed feature selection method proves better separability. Regarding repeatability and robustness, the proposed method is always converge to same/similar solution regardless of the used randomness.

Acknowledgment

This work was partially supported by the IPROCOM Marie Curie initial training network, funded through the People Programme (Marie Curie Ac-



(a) Small Initialization



(b)Large Initialization

Figure 5: Std measure average over all the data sets for the different optimizers

tions) of the European Union's Seventh Framework Programme FP7/2007-2013/ under REA grant agreement No. 316555. This fund only apply for one co-authors (Hossam M. Zawbaa).

References

[1] B. Chizi, L. Rokach, O. Maimon, "A Survey of Feature Selection Techniques", Encyclopedia of Data Warehousing and Mining, Second Edition, IGI Global, pp. 1888-1895, 2009.

[2] G. Chandrashekar, F. Sahin, "A survey on feature selection methods", Computers and Electrical Engineering, Vol. 40, No. 1, pp. 16-28, 2014.

Table 10: The Wilcoxon test for the average fitness obtained by the different optimizers

Dataset	Normal Initialization		Small Initialization		Large Initialization	
	bGWO1	**bGWO2**	**bGWO**	**bGWO2**	**bGWO**	**bGWO2**
GA	0.056	0.151	0.548	1	0.008	0.008
PSO	0.056	0.032	0.008	0.008	0.69	0.008

[3] D. Bell, H. Wang, "A Formalism for Relevance and its Application in Feature Subset Selection", Machine Learning, Vol. 41, No. 2, pp. 175195, 2000.

[4] R. O. Duda, P. E. Hart, D. G. Stork, "Pattern Classification" (2nd Edition), Wiley-Interscience, 2000.

[5] R. Y. M. Nakamura, L. A. M. Pereira, K. A. Costa, D. Rodrigues, J. P. Papa, X-S Yang, "BBA: A binary bat algorithm for feature selection", In Graphics, Patterns and Images (SIBGRAPI), pp. 291-297, 2012.

[6] S. Shoghian, M. Kouzehgar, "A Comparison among Wolf Pack Search and Four other Optimization Algorithms", World Academy of Science, Engineering and Technology, Vol. 6, 2012.

[7] J. Holland, "Adaptation in natural and artificial systems", Ann Arbor, MI: university of Michigan Press, 1975.

[8] R. C. Eberhart, J. Kennedy, "A New Optimizer Using Particle Swarm Theory", Proceeding of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, pp. 39-43, 1995.

[9] D. Karaboga, B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm", Journal of Global Optimization, Vol. 39, No. 3, pp. 459-471, 2007.

[10] X. S. Yang, "Engineering optimizations via nature-inspired virtual bee algorithms", In: Lecture notes in computer science, Springer (GmbH), pp. 317323, 2005.

[11] K. Sundareswaran, V. T. Sreedevi, "Development of novel optimization procedure based on honey bee foraging behavior", IEEE International conference on systems, man and cybernetics, pp. 12201225, 2008.

[12] R. Akbari, A. Mohammadi, K. Ziarati, "A novel bee swarm optimization algorithm for numerical function optimization", Commun Nonlinear Sci Numer Simulat, pp. 3142-3155, 2010.

[13] X. L. Li, Z. J. Shao, J. X. Qian, "An Optimizing Method Based on Autonomous Animates: Fish-swarm Algorithm", Methods and Practices of System Engineering, pp. 3238, 2002.

[14] J. Kennedy, R. C. Eberhart, "A discrete binary version of the particle swarm algorithm", IEEE International Conference on Systems, Man, and Cybernetics, Vol. 5, pp. 4104-4108, 1997.

[15] H. A. Firpi, E. Goodman, "Swarmed feature selection", Proceedings of the 33rd Applied Imagery Pattern Recognition Workshop, Washington, DC, USA, IEEE Computer Society, pp. 112118, 2004.

[16] H. Nezamabadi-pour, M. Rostami-shahrbabaki, M. M. Farsangi, "Binary Particle Swarm Optimization: challenges and New Solutions", The Journal of Computer Society of Iran (CSI) On Computer Science and Engineering (JCSE), Vol. 6, No. (1-A), pp. 21-32, 2008.

[17] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, "BGSA: binary gravitational search algorithm", Natural Computing, Vol. 9, pp. 727745, 2010.

[18] R. Y. M. Nakamura, L. A. M. Pereira, K. A. Costa, D. Rodrigues, J. P. Papa, X.-S. Yang, "BBA: A Binary Bat Algorithm for Feature Selection", IEEE XXV SIBGRAPI Conference on Graphics, Patterns and Images, pp. 291-297, 2012.

[19] H. R. Kanan, K. Faez, S. M. Taheri, "Feature selection using ant colony optimization (ACO): a new method and comparative study in the application of face recognition system", in Proceedings of the 7th industrial conference on Advances in data mining: theoretical aspects and applications, Berlin, Heidelberg, Springer-Verlag, pp. 6376, 2007.

[20] J. Huang, Y. Cai, X. Xu, "A hybrid genetic algorithm for feature selection wrapper based on mutual information", Pattern Recognition Letters, Vol. 28, No. 13, pp. 18251844, 2007.

[21] H. Banati, M. Bajaj, "Fire Fly Based Feature Selection Approach", International Journal of Computer Science Issues, Vol. 8, No. 4, pp. 473480, 2011.

[22] S. Mirjalili, S. M. Mirjalili, A. Lewis, "Grey Wolf Optimizer", Advances in Engineering Software, Vol. 69, pp. 46-61, 2014.

[23] Y. Zhang, S. Wang, P. Phillips, G. Ji, "Binary PSO with mutation operator for feature selection using decision tree applied to spam detection", Original Research Article Knowledge-Based Systems, Vol. 64, pp. 22-31, July 2014.

[24] L. Y. Chuang, S. W. Tsai, C. H. Yang, "Improved binary particle swarm optimization using catfish effect for feature selection", Original Research Article Expert Systems with Applications, Vol. 38, No. 10, pp. 12699-12707, 15 September 2011.

[25] S. Mirjalili, G. G. Wand, L. Coelho, "Binary optimization using hybrid particle swarm optimization and gravitational search algorithm", Neural Computing and Applications, Vol. 25, No. 6, pp. 1423-1435, 2014.

[26] K. Suresh, N. Kumarappan, "Hybrid improved binary particle swarm optimization approach for generation maintenance scheduling problem", Swarm and Evolutionary Computation, Vol. 9, pp. 69-89, 2013.

[27] Md. Abul Kalam Azad, Ana Maria A.C. Rocha, Edite M.G.P. Fernandes, "Improved binary artificial fish swarm algorithm for the 01 multidimensional knapsack problems", Swarm and Evolutionary Computation, Vol. 14, pp. 66-75, 2014.

[28] L. Y. Chuang, H. W. Chang, C. J. Tu, C. H. Yang, "Improved binary PSO for feature selection using gene expression data", Computational Biology and Chemistry, Vol. 32, pp. 29-38, 2008.

[29] A. Frank, A. Asuncion, UCI Machine Learning Repository, 2010.

[30] Kennedy, J.; Eberhart, R., "Proceedings of IEEE International Conference on Neural Networks, IV.", pp. 1942-1948, 2009.

[31] A. E. Eiben, P. -E. Raue, Zs. Ruttkay, "Genetic algorithms with multi-parent recombination", PPSN III: Proceedings of the International Conference on Evolutionary Computation, The Third Conference on Parallel Problem Solving from Nature, pp. 78-87, 1994.

[32] P. E. H. R. O. Duda, D. G. Stork, "Pattern Classification", Wiley-Interscience Publication, 2001.

[33] F. Wilcoxon, "Individual Comparisons by Ranking Methods", Biometrics Bulletin, Vol. 1, No. 6., pp. 80-83, 1945.