Feature Selection using Hyper Learning-Based Binary Political Optimizer Algorithm

Mehroze Khan, Irfan Younas, Maryam Bashir (maryam.bashir@nu.edu.pk)

FAST School of Computing, National University of Computer and Emerging Sciences, Lahore, Pakistan

Abstract.

Feature selection is a crucial preprocessing step in machine learning that involves selecting a subset of relevant features to improve the accuracy and efficiency of classification algorithms. In this study, we propose a novel algorithm called Hyper Learning-Based Binary Political Optimizer (HLBPO) for feature selection. HLBPO is an enhanced version of the Political Optimizer algorithm that incorporates hyper-learning to improve its search behavior and avoid local minima solutions. We evaluate the performance of HLBPO on 21 classification datasets and compare it with nine state-of-the-art feature selection algorithms. Our findings demonstrate that HLBPO outperforms the other algorithms in terms of reducing the number of features and improving classification accuracy. The low feature selection ratio of HLBPO suggests that it selects only the most essential features, resulting in increased accuracy and reduced processing time. Our study highlights the potential of HLBPO as an efficient and effective feature selection method. The study proposes a novel algorithm called Hyper Learning-Based Binary Political Optimizer (HLBPO) for feature selection, which is an enhanced version of the Political Optimizer algorithm that incorporates hyper-learning to improve its search behavior and avoid local minima solutions. The performance of HLBPO is evaluated on 21 classification datasets and compared with nine state-of-the-art feature selection algorithms. The findings demonstrate that HLBPO outperforms the other algorithms in terms of reducing the number of features and improving classification accuracy. The low feature selection ratio of HLBPO suggests that it selects only the most essential features, resulting in increased accuracy and reduced processing time. The study highlights the potential of HLBPO as an efficient and effective feature selection method.

Keywords: Feature selection, Hyper learning, Binary Political Optimizer, Metaheuristics, Optimization, Classification accuracy

1 Introduction

Rapid data and information expansion has inspired data mining and data science researchers to develop innovative data preprocessing approaches. A significant issue is the huge number of data dimensions or characteristics. The majority of these characteristics are noisy, which hinders the effectiveness of machine learning algorithms. Feature selection is the process of lowering data dimensionality to enhance the prediction accuracy and time complexity of machine learning algorithms. The quantity of the data is reduced by selecting just the most vital elements and discarding the rest. As a consequence, algorithm processing times and precision increase. Preprocessing is a crucial stage in data mining, machine learning, and statistics. There are several applications for feature selection in the medical field, bioinformatics, social media, and multimedia retrieval.

In the last ten years, several feature selection strategies have been presented [49, 50]. There are two phases to the feature selection process. Selection of a subset of characteristics and assessment of that subset. A search technique is used to choose a subset of features, which is then assessed using a machine-learning classifier. All feature selection approaches fall into two categories: filer-based and wrapper-based strategies. Filer-based algorithms [51] analyze the chosen features based on their data-specific properties. Wrapper-based approaches [48] assess the chosen feature subset using the prediction accuracy of a classifier. Although filter-based approaches are more economical in terms of temporal complexity, wrapper-based methods perform better in terms of classifier precision.

The selection of the optimal subset of features is an optimization problem. As the number of dimensions rises, so does the size of the search space. The number of alternative solutions is often exponential, making the issue NP-hard when solved using exact algorithms. NP-hard issues are solved via approximate algorithms that approximate the ideal answer. In the last decade, metaheuristics have shown remarkable performance on several NP-hard issues. Meta-heuristics are an algorithmic framework that provides the fundamental rules for developing heuristic algorithms. Due to its adaptability, nature-inspired metaheuristics have proved beneficial for tackling optimization challenges [1]. These algorithms draw inspiration from nature and use this knowledge to tackle real-world challenges. Genetic Algorithm [2], Memetic Algorithm [3], Genetic Programming [4], and Cellular Evolutionary Algorithm [5] are a few of the evolution-inspired algorithms that have been used lately. Swarm intelligence is the behaviour of living creatures as a group. It involves looking for food, dwelling in colonies, hunting, mating, and developing. Ant Colony Optimization [6], Whale Optimization Algorithm [7], Grey Wolf Optimization Algorithm [8], Dragonfly Algorithm [9], and Bee Colony Algorithm [10] are well-known algorithms. Carrier frequency offset (CFO) Algorithm [11], Asynchronous Partial Overlay (APO) Algorithm [12], and Gravitational Search Algorithm (GSA)

[13] are examples of algorithms influenced by physics. Human League Championship Algorithm [14], Anarchic Society Optimization [15], and Grammatical Evolution Algorithm [16] are behavior-inspired algorithms. Because of its superior performance, several researchers have used nature-inspired metaheuristics to the issue of feature selection in recent years. Basset et al. [22] suggested a two-phase mutation grey wolf optimizer technique for feature selection. For dimensionality reduction, Hussien et al. [31] suggested a binary whale optimization technique. In the section on related work, more algorithms are explored.

Political Optimizer (PO) [17] is a socio-inspired algorithm recently presented for handling optimization problems. It is inspired by human behavior in politics and imitates candidates' party-switching behavior. It includes the essential aspects of an electoral process, such as the founding of several political parties, the assignment of different seats to candidates, the selection of leaders from each party and victors from each district, party switching, and parliamentary affairs. This algorithm's distinguishing characteristics include the dual function of each potential solution. The solution functions as both a member of a political party and a candidate in an election, allowing every solution to be updated twice and increasing the likelihood of improvement. In addition, it presents a novel technique for revising the stance of candidates that integrates the behaviors that politicians have learnt in the past. PO outperformed the performance of fifteen other algorithms because to its superior convergence speed and exploration.

The choice of the Political Optimizer (PO) algorithm as the starting point for the proposed approach was based on several key factors that make it a suitable foundation for further enhancement in the field of feature selection like The PO algorithm is inspired by the political system, which introduces a unique and innovative perspective to optimization problems. By mimicking the dynamics of political processes, the PO algorithm offers a different approach to exploration and exploitation in search spaces, which can lead to novel solutions. The PO algorithm is a relatively recent addition to the metaheuristics landscape, introduced in 2020. Its novelty and potential for further improvement make it an attractive choice for building upon and enhancing its capabilities. The PO algorithm incorporates a recent past-based position updating strategy that balances exploration and exploitation effectively. This feature is crucial in optimization problems, especially in feature selection, where finding the right balance between exploring new solutions and exploiting known solutions is essential. It has shown promising performance in solving feature selection problems. By leveraging its strengths and enhancing its capabilities through hyper learning, the proposed approach aims to further improve its performance and effectiveness in identifying optimal feature subsets. While there are many metaheuristics algorithms available today, the unique socio-inspired nature of the PO algorithm presents an opportunity for innovation and the development of novel techniques. By starting with a base

algorithm like PO and introducing hyper learning concepts, the proposed approach aims to push the boundaries of feature selection optimization and achieve results beyond what current algorithms offer.

Later, a Binary Political Optimizer (BPO) [18] method was devised for handling discrete issues. The PO algorithm handles continuous problems. BPO demonstrated extremely excellent convergence and accuracy in comparison to eight other algorithms on nine datasets, however the method is very exploitative. Hence, the algorithm might get stuck in local optima. Hence, BPO is at times unable to identify globally optimum solutions. To address this issue, we offer a hyper learning-based binary political optimizer (HLBPO) method that can locate the optimum global solutions for the feature selection problem. The objective is to identify the most representative collection of features for enhancing the classification precision of machine learning algorithms. In comparison to nine state-of-the-art algorithms for feature selection, our suggested HLBPO method exhibited superior performance. The following are the primary contributions of this study:

• A new algorithm HLBPO is proposed for feature selection challenges.

• Enhancement of binary PO's exploratory rate. To enhance the exploratory behavior of the binary PO method, hyper learning is used.

• Performance comparison between HLBPO and nine state-of-the-art algorithms on twenty-one low, medium, and high dimensional datasets.

• Assessment of HLBPO's superiority over nine state-of-the-art feature selection algorithms.

The motivation behind the proposed research lies in addressing the challenges posed by feature selection in machine learning and optimization problems. The "beyond state-of-the-art" aspect of the proposed study is the development of the Hyper Learning-Based Binary Political Optimizer (HLBPO) algorithm, which aims to revolutionize feature selection by combining the strengths of hyper learning, binary optimization, and political-inspired optimization. This algorithm represents a significant advancement beyond the current state-of-the-art in feature selection by offering a novel approach that outperforms existing algorithms in terms of convergence speed, exploration, and classification accuracy. By integrating the principles of hyper learning and political optimization, the HLBPO algorithm seeks to achieve superior performance in identifying the most informative feature subset while reducing computational complexity. This innovative approach represents a leap forward in the field of feature selection and has the potential to significantly enhance the efficiency and effectiveness of machine learning and optimization tasks.

The organization of the rest of the article is as follows: Section 2 presents the literature review of all related socio-inspired algorithms for solving feature selection problems. Section 3 gives an overview of the Political Optimizer Algorithm. Section 4 presents the proposed hyper learning-based binary political optimizer algorithm (HLBPO). Section 5 gives details of experiments, results, and comparisons with other algorithms for feature selection. Finally, Section 6 concludes the paper and gives directions for future work.

2 Related Work

In order to improve the performance and outcomes of feature selection, hybrid algorithms are often used. The advantage of a hybrid algorithm is that it combines the advantages of two algorithms. One such program developed a hybrid binary optimization algorithm [20] by combining the particle swarm and binary grey wolf algorithms. The performance of the hybrid algorithm improved, but the feature selection ratio fell. It employs the wrapper approach of k closest neighbor to discover optimum solutions and has been evaluated on 18 datasets. Combining the simulated annealing approach with the whale optimization-algorithm, another hybrid algorithm [21] was employed to provide a new technique for feature selection. First, the whale optimization method seeks out the most promising locations, and then the simulated annealing process boosts the regions' exploitation. This algorithm's classification accuracy is greater than that of existing wrapper-based approaches, as shown by a comparison with 18 more datasets.

A grey wolf optimization method [22] that uses two-phase mutation to classify and resolving the issue of feature selection was presented. The first step of mutation decreases the feature while preserving classification accuracy, but the second phase provides key traits that improve classification. The approach employs the k-nearest neighbor wrapper method, and its performance is measured against 35 different datasets. A whale optimization method [23] with two binary invariants was proposed. In lieu of a random operator, roulette and tournament selection are employed for the first invariant, and crossover and mutation operators are used for the second invariant. The ant colony algorithm [25] was developed using both the filter approach and the wrapper method. This approach evaluates the subset using the filter technique rather than the wrapper method, reducing the computational complexity. It also has a memory that recalls the finest ants. For the multilabel classification issue, MMFS [19] was presented as a feature selection method. The convergence rate and variety of NSGA-many-objective III's algorithm were enhanced by the addition of two archives. The uniform crossover and mutation operators were also suggested to enhance the exploration. Using eleven distinct datasets with various labels, the method was evaluated and showed strong classification

results, eliminated unnecessary features, and balanced numerous goals; nevertheless, the cost of computing the wrapper features is high. An enhanced differential evolution method [48] for feature selection using the wrapper-filer feature subset selection strategy was presented. Fuzzy estimators were used to initialize a small number of solutions with a preset number of best features, which assisted in locating the best features from the global search space. The adaptive differential evolution method with linear population size reduction (LSHADE) [45] is another example of a differential evolution-based approach for feature selection. A chaotic crow search algorithm (CCSA) [44] was suggested and evaluated on 20 benchmark datasets for feature selection. It outperformed several current feature selection methods.

Text clustering is used to categorize comparable content into a single group. The clustering issue is suggested to be solved via a particle swarm-based optimization algorithm [24]. This approach introduces a new subset of the most informative significant attributes. It outperforms other text clustering methods such as the Genetic Algorithm and the Harmony Search Algorithm.

Intrusion detection is one of the greatest challenges in network security. Research [26] addressed this issue by building a safe and efficient intrusion detection system that takes into account all of the system's essential characteristics. The system was inspired by an ant colony system that determines the ideal characteristics and was successful enough to detect infiltration attempts.

To categorize data, feature selection is very commonly utilized in the area of medical research. A social spider algorithm [27] inspired on the social behavior of spiders is one such application. Abdominal CT is used to identify liver tumors. This technique identifies the most promising and optimum search space locations. Exploration and exploitation are accomplished by dividing the population into male and female categories, which also increases the probability of obtaining global optimality.

Several metaheuristic algorithms provide continuous results even though feature selection is a binary optimization issue. Hence, much recent research has turned these continuous-valued algorithms into binary ones [28] in order to lower the space and computational complexity of algorithms. The space-saving binary Bat Algorithm [29] is presented to overcome the issue. The s-shaped transfer function is used to convert continuous data to binary values. Moreover, the binary algorithms may be integrated with other algorithms. Combining a binary particle swarm technique with simulated annealing, the BSAPSO algorithm [30] increases convergence speed. Another study [31] introduced two binary versions of the whale optimization algorithm utilizing two transfer functions, one version utilizing an s-shaped transfer function and the other version utilizing a V-shaped transfer function and demonstrates

greater efficiency than other binary socio-inspired algorithms such as Binary Particle Swarm Optimizer [32], Binary Ant Lion Optimizer [8], and Binary Dragonfly algorithm [33]. Other binary evolutionary algorithms for feature selection include the hyper learning binary dragonfly algorithm (HLBDA) [40], the binary artificial bee colony (BABC) [41], the binary dragonfly algorithm (BDA) [33], the binary coyote optimization algorithm (BCOA) [46], the binary particle swarm optimization (BPSO) [42], and the binary multiverse optimizer (BMVO) [20].

Despite numerous studies on evolutionary algorithms for feature selection, there is always opportunity for improvement. In this paper, we present a binary political optimizer based on hyper learning for feature selection. The next section describes the political optimizer algorithm [17] in depth.

This section provides an overview of current research on the use of metaheuristics for feature selection. Metaheuristics are algorithms that can optimize situations with unknown solutions that are complicated. Using the advantages of two algorithms, hybrid algorithms are often used to improve the performance and outcomes of feature selection. Many metaheuristics-based hybrid algorithms, such as the binary optimization algorithm, simulated annealing method with the whale optimization algorithm, grey wolf optimization algorithm, and ant colony algorithm, have been developed. In order to lower the computing complexity of the algorithms, continuous-valued algorithms have been transformed into binary ones. Several binary algorithms, such as the binary Bat Algorithm, BSAPSO algorithm, binary versions of the whale optimization algorithm, the binary dragonfly method, and the binary coyote optimization algorithm, have been developed. Even though these algorithms have shown enhanced performance, there is still potential for advancement, and this work seeks to address that. The section also focuses briefly on the application of metaheuristics to issues including text clustering, intrusion detection, and medical categorization.

3 Political Optimizer

Political Optimizer (PO) Algorithm [17] proposed in the year 2020, is a socio-inspired algorithm for solving the feature selection problem. This algorithm is inspired by the political system of our social environment. It incorporates different phases of politics. The exploration and exploitation are performed by using the recent past-based position updating strategy (RPPUS) which allows the algorithm to learn from the previous election. It learns by improving the weak performing solutions to better solutions like party leaders and by comparing the solutions with winners of constituencies and updating their positions to them. The balance in the exploration and exploitation is

formed by the party-switching phase. Fitness evaluation of the candidate solutions is done using the election phase. To introduce exploitation and convergence in the algorithm, the parliamentary phase is incorporated. The flowchart of the PO algorithm is given in Figure 1.

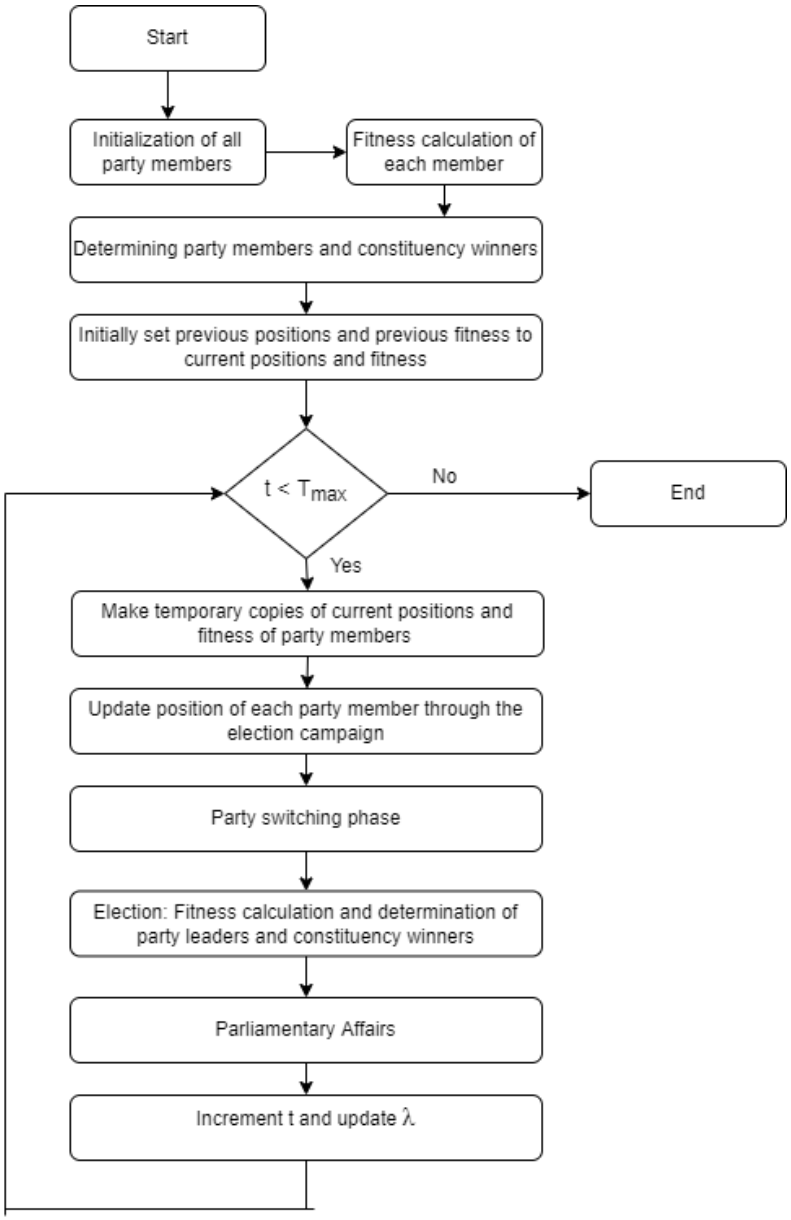

Fig. 1. Flowchart of PO Algorithm [17]

There are five major phases in this algorithm. These are explained in the following.

3.1 Party Formation and Constituency Allocation

The total population P is distributed into n political parties as expressed in Eq. (1). Each party P_i has n members in it as expressed in Eq. (2). The j^{th} member of an i^{th} party is represented as p_i^j. The members are represented by using a d-dimensional vector which is basically the number of input variables. The k^{th} dimension is represented as $p_{i,k}^j$ which means j^{th} member of i^{th} political party in the k^{th} dimension as expressed in Eq. (3).

$$P = \{P1, P2, P3, \dots, Pn\} \quad (1)$$

$$Pi = \{p_i^1, p_i^2, p_i^3, \dots, p_i^n\} \quad (2)$$

$$p_i^j = \{p_{i,1}^j, p_{i,2}^j, p_{i,3}^j, \dots, p_{i,d}^j\}T \quad (3)$$

The constituencies are divided into n different constituencies as shown in Eq. (4). The jth member of each party competes for an election in the jth constituency as shown in Eq. (5). This logical division is shown in Figure 3.1 which is taken from [17].

$$C = \{C1, C2, C3, \dots, Cn\} \quad (4)$$

$$C_j = \{p_1^j, p_2^j, p_3^j, \dots, p_n^j\} \quad (5)$$

The set of selected party leaders and constituency winners are represented as shown in Eq. (6) and Eq. (7). These sets are calculated in the election phase.

$$P^* = \{p_1^*, p_2^*, p_3^*, \dots, p_n^*\} \quad (6)$$

$$C^* = \{c_1^*, c_2^*, c_3^*, \dots, c_n^*\} \quad (7)$$

3.2 Election Campaign

In the election campaign, the solutions improve their chances of getting elected by improving themselves based on three aspects. First, the past learning experience is used to update the position of solutions by using RPPUS strategy as shown in Eq. (8) and Eq. (9). Second, the position of solutions is updated with respect to the leaders of their party. Finally, the position of candidate solutions is updated with respect to the constituency winners. This increases the chance of obtaining better solutions. The current fitness f (p_i^j(t)) is compared to the previous fitness f (p_i^j(t-1)) and if there is an improvement in the fitness, then Eq. (8) is used to update position otherwise Eq. (9) is

used. In this equation, r symbolizes a random number in the range of 0 to 1 and m* holds the kth dimension of both, party leader and constituency winner.

Eq. (8):-

$$p_{i,k}^{j}(t+1)=$$

$$\begin{cases} m^* + r\left(m^* - p_{i,k}^{j}(t)\right) & if\ p_{i,k}^{j}(t-1) \le p_{i,k}^{j}(t) \le m^*\ or\ p_{i,k}^{j}(t-1) \ge\ p_{i,k}^{j}(t) \ge m^* \\ m^* + (2r-1)\left|m^* - p_{i,k}^{j}(t)\right| & if\ p_{i,k}^{j}(t-1) \le m^* \le p_{i,k}^{j}(t)\ or\ p_{i,k}^{j}(t-1) \ge m^* \ge\ p_{i,k}^{j}(t) \\ m^* + (2r-1)\left|m^* - p_{i,k}^{j}(t-1)\right| & if\ m^* \le\ p_{i,k}^{j}(t-1) \le p_{i,k}^{j}(t)\ or\ m^* \ge\ p_{i,k}^{j}(t-1) \ge\ p_{i,k}^{j}(t) \end{cases}$$

Eq. (9):-

$$p_{i,k}^{j}(t+1)=$$

$$\begin{cases} m^* + (2r-1)\left|m^* - p_{i,k}^{j}(t)\right| & if\ p_{i,k}^{j}(t-1) \le p_{i,k}^{j}(t) \le m^*\ or\ p_{i,k}^{j}(t-1) \ge\ p_{i,k}^{j}(t) \ge m^* \\ p_{i,k}^{j}(t-1) + r\left(p_{i,k}^{j}(t) - p_{i,k}^{j}(t-1)\right) & if\ p_{i,k}^{j}(t-1) \le m^* \le p_{i,k}^{j}(t)\ or\ p_{i,k}^{j}(t-1) \ge m^* \ge\ p_{i,k}^{j}(t) \\ m^* + (2r-1)\left|m^* - p_{i,k}^{j}(t-1)\right| & if\ m^* \le\ p_{i,k}^{j}(t-1) \le p_{i,k}^{j}(t)\ or\ m^* \ge\ p_{i,k}^{j}(t-1) \ge\ p_{i,k}^{j}(t) \end{cases}$$

3.3 Party Switching

This phase is incorporated to balance out exploration as well as exploitation. There is a parameter named party switching rate represented as λ that starts form the value λ_{max} and then decreases to zero. All the members p_i^j are selected with the probability of λ and then switched with the least fit member of a randomly selected party P_r which is shown in Eq. (10).

$$q = \underset{1 \le j \le n}{\text{argmax}} f(p_r^j) \qquad (10)$$

3.4 Election

In this phase, the fitness of each member that is competing in the election for a constituency is evaluated and the fittest member is chosen as the winner of the constituency.

In this phase, the fitness of each member that is competing in election for a constituency is evaluated and the fittest member is chosen as the winner of constituency as shown in Eq. (11).

$$q = \underset{1 \le i \le n}{\text{argmin}} f(p_i^j) \qquad (11)$$

$$c_j^* = p_q^j$$

The party leaders are selected using Eq. (12). The fittest member of a party is chosen as its leader.

$$q = \underset{1 \le j \le n}{\operatorname{argminf}}\left(p_i^j\right) \qquad \forall i \in \{1, \ldots, n\} \qquad (12)$$

$$p_i^* = p_i^q$$

3.5 Parliamentary Affairs

After the formation of the government, the winners of constituencies and the leaders of parties are selected. Then every member of the parliament is updated to some other randomly selected parliament member. If there is an improvement in the solution's fitness, the solution is updated otherwise the solution is not updated and remains the same.

The election campaign, party switching, election, and parliamentary affairs phase provide optimization by performing different local and global searches. The election campaign and parliamentary affairs provide exploitation by searching in the most promised regions of search space which allows the PO to show superior performance.

PO algorithm is used to solve continuous problems. A binary political optimizer [18] has been proposed to solve the feature selection problem. In this study, we propose a hyper learning-based binary political optimizer for the feature selection problem. Its details are given in the next section.

4 Hyper Learning BASED Binary Political Optimizer Algorithm

In this article, a Hyper Learning-Based Binary Political Optimizer Algorithm (HLBPO) is introduced to solve the feature selection problem. This algorithm utilizes the concept of updating the position of candidate solutions to personal best and global best solutions. In the PO algorithm, the position of candidates was updated using the personal best solutions (party leaders). By introducing the concept of position updating to both personal best solution and global best solution, the solutions, and their searching behavior are expected to improve. The proposed HLBPO algorithm is shown in Algorithm 1.

Algorithm 1: Hyper Learning Binary Political Optimizer Algorithm (HLBPO)

Input: n (number of party members, political parties and constituencies), λ_{max} (The upper limit of the party-switching rate), T_{max}(total number of iterations), plr (personal learning rate), glr (global learning rate)

Output: final population P(T_{max})

Party Formation and Constituency Allocation (P, n)

Initialization (n)

Election (n)

t = 1;

$\lambda = \lambda_{max}$;

while t ≤ T_{max}do

P(t − 1) = P;

f (P(t − 1)) = f (P);

foreach Pi ∈ P do

foreach $p_i^j \in$ Pi do

p_i^j = ElectionCampaign(p_i^j , p_i^j (t − 1), p_i^*, c_j^*);

end

end

PartySwitching (P, λ);

Election (n);

ParliamentaryAffairs (C^*, P);

Hyper Learning Position Updating Strategy (n, plr, glr)

$\lambda = \lambda - \lambda_{max}/T_{max}$;

t = t + 1;

end

4.1 Initialization

The HLBPO first sets the parameters and then initializes a random population of candidate solutions with values 0 and 1. If the value is 0 it means that the particular feature is not selected and if the corresponding value is 1 then the feature is selected. Initialization steps are shown in Algorithm 2.

Algorithm 2: Initialization (n)

foreach Pi ∈ P do

  foreach $p_i^j \in$ Pi do

    r ← random number in the interval [0, 1]

    if r ≤ 0.5 then

      $p_i^j = 0$

    else

      $p_i^j = 1$

    end

  end

end

Table 1 shows the parameter setting of HLBPO. The number of solutions is set to be 10 and the total iterations are set to be 100. Two new parameters are used namely plr and glr which need a very careful adjustment.

Table 1. Parameter setting of HLBPO

Parameters	**Values**
Number of Candidate Solutions (N)	10
Total Iterations (T)	100
Dimensions (D)	Number of features in each dataset
Global Learning Rate (glr)	0.7
Personal Learning Rate (plr)	0.4

4.2 Election

In the election phase, the first fitness of each member is calculated using the fitness function. After calculating the fitness function, the sets of party leaders and constituency winners are formed. In the next step, the global best solution is recorded. The global best solution is the one whose fitness value is least among all the candidate solutions. The election phase steps are shown in Algorithm 3.

Algorithm 3: Election (n)

foreach Pi ∈ P do

  foreach p_i^j ∈ Pi do

    Calculate fitness of p_i^j using Eq. (18)

  end

end

Compute set of party leaders P^* using Eq. (12)

Compute set of constituency winners C^* using Eq. (11)

Store the position and fitness of global best solution

End

4.3 Hyper Learning Position Updating Strategy

This strategy enables the solutions to learn from both personal best solutions as well as global best solutions in the course of their search phase. In the proposed technique, the position of candidates is updated using the following Eq. (13) and (14).

$$p_i^j(t+1) = \begin{cases} \overline{p_i^j} & 0 \le r_1 < \text{plr} \\ pb_i^j\ (\text{t}) & \text{plr} \le r_1 < \text{glr} \\ gb\ (t) & \text{glr} \le r_1 \le 1 \end{cases} \quad (13)$$

$$\overline{p_i^j} = \begin{cases} 1 - p_i^j(t) & r_2 < \text{P}\ (p_i^j\ (\text{t})) \\ p_i^j(t) & r_2 \ge \text{P}\ (p_i^j\ (\text{t})) \end{cases} \quad (14)$$

Here p_i^j represents the position of the ith member in jth political party, pb represents the position of the personal best candidate, gb is the position of the global best solution, t is the iteration number, r_1 and r_2 are two random numbers generated in the interval of [0,1] and plr and glr represents personal and global learning rates and their value is constant which lies in the interval of [0,1]. Hyper learning is shown in Algorithm 4.

Algorithm 4: Hyper Learning Position Updating Strategy (n, plr, glr)

```
foreach Pi ∈ P do
    foreach p_i^j ∈ Pi do
        TF = Calculate probability using Transfer Function in Eq. (16)
        r_1 ← random number in the interval [0, 1]
        if 0 ≤ r_1 < plr then
            Update position using Eq. (14)
        elseif plr ≤ r_1 < glr then
            Update position with respect to personal best solution
        elseif glr ≤ r_1 ≤ 1
            Update position with respect to global best solution
        end
    end
end
```

The plr and glr play a very significant role during the learning process. If their values are kept too low, then the search region will be around personal and global best solutions, so the solutions are more prone to stuck in the local optima. If the values of plr and glr are kept too high, then the position updating strategy will become similar to the binary PO. So, the choice of these parameters is very important.

Feature selection is a binary optimization problem. In the PO algorithm, the fitness of the members decides their positions in the search space, but the fitness values are continuous which should be transformed into binary values to use for binary feature selection problem because in the feature selection technique the search space is a Boolean n-dimensional matrix.

To generate the probability of changing the position of candidate solutions, transfer functions are used. There are two main categories of transfer function: S and V-shaped as shown in Figure 2 and Figure 3 [36].

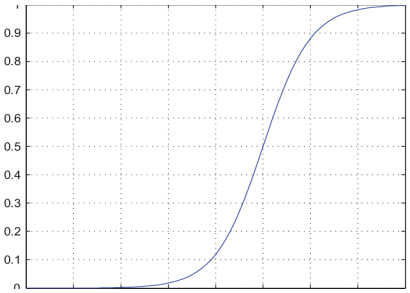

Fig. 2. S-shaped Transfer Function [34]

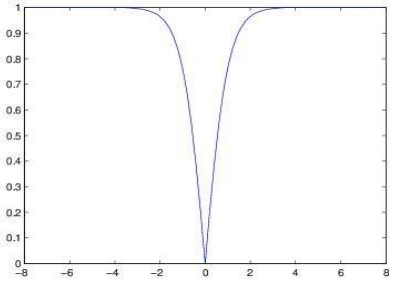

Fig. 3. V-shaped Transfer Function [35]

In this algorithm, the transfer function used to generate the probability is a V-shaped transfer function that computes the altering probability of the candidate's position. This transfer function does not restrict the candidates to choose a value of either 0 or 1. This allows the algorithm to perform high exploration and find more promising regions in the search space. The transfer function used in HLBPO is shown in Eq. (16).

$$P\left(p_i^j(t)\right) = TF\left(p_i^j(t)\right) \qquad (15)$$

$$TF(x) = \left|\frac{x}{\sqrt{x^2+1}}\right| \qquad (16)$$

4.4 Fitness Evaluation

The fitness evaluation of all the candidate solutions will be done using an objective function. The purpose of using an objective function is to evaluate the quality of solutions. The objective function used in the proposed algorithm minimizes the features selection ratio and classification error rate. The objective function [8] is given in Eq. (17).

$$Fitness = \alpha ER + \beta\left(\frac{|SF|}{|TF|}\right) \qquad (17)$$

Here ER represents the error rate of classification in a specific classifier. The classifier used in this study is K nearest neighbor also known as the KNN classifier. |SF| is the length of the subset of selected features and |TF| is the length of total features in the dataset. α and β are two parameters. α impacts the error of classification with a value in the range of [0, 1] and β impacts the feature size with a value of β = (1-α).

In the first step, the datasets are partitioned into two sets; training and validation sets with the help of a stratified 10-fold cross-validation technique. The training set is used to train the model and the validation set is the set of samples that are held back during the formation of the training set to evaluate the selected features. The reason for using the KNN algorithm is because it is simple and requires no prior training for making predictions so new data can be added very easily [37].

4.5 Termination

After every iteration, the position of the candidate solutions is updated. After a pre-defined number of iterations, the algorithm terminates.

5 Experiments and Results

This section presents details of datasets, experimental setup, and results of this study.

5.1 Datasets

The performance of the proposed HLBPO algorithm is evaluated by running it on twenty-one datasets taken from UC Irvine Repository for Machine Learning [38] and Arizona State University [39]. These datasets comprise of a different number of features, instances, and dimensions. The details of these datasets are provided in Table 2.

Table 2. Description of datasets

No.	Dataset Name	No. of Instances	No. of Features	Dimensions
1	Ionosphere	351	34	Medium
2	TOX_171	171	5748	High
3	Colon	62	2000	High
4	Leukemia	72	7070	High
5	Hepatitis	155	19	Low
6	Dermatology	366	34	Medium
7	ILPD	583	10	Low

8	Lung Discrete	73	325	Medium
9	Glass	214	10	Low
10	Horse Colic	368	27	Low
11	SPECT Heart	267	22	Low
12	SCADI	70	205	Medium
13	Lymphography	148	18	Low
14	Zoo	101	17	Low
15	Arrhythmia	452	279	Medium
16	Soybean	307	35	Medium
17	LSVT	126	310	Medium
18	Musk 1	476	168	Medium
19	Primary Tumor	339	17	Low
20	Seeds	210	7	Low
21	Libras Movement	360	90	Medium

5.2 Experimental Setup

To evaluate the algorithm performance, five different evaluation metrics are considered namely best fitness, mean fitness, standard deviation, feature selection ratio, and classification accuracy. Ten independent runs are performed and then the averages of results are calculated. The proposed HLBPO is implemented in MATLAB R2019a and it is executed on a Core i3-4030U with 4GB RAM.

5.3 Results and Comparison

The performance of HLBPO is compared with 9 other well-known feature selection algorithms. These algorithms are HLBDA [40], BABC [41], BDA [33], BPSO [42], BMVO [20], CMAES [43], CCSA [44], LSHADE [45] and BCOA [46]. The number of solutions and the total iterations are kept the same for each algorithm. Table 3 shows the parameter settings of compared algorithms. For BDA and HLBDA all the parameter values are kept identical

to the original paper. In BPSO algorithm, the inertia weight is linearly decreasing from 0.9 to 0.4 and the acceleration factors are set with the value 2. In CCSA, the awareness probability is 0.1, and flight length is 2. The number of solutions for each algorithm is 10 and the maximum number of iterations is 100.

Table 3. Parameter settings of compared algorithms

Algorithm	Parameter	Value
BDA	All controlling parameters	Identical to original paper
HLBDA	All controlling parameters	Identical to original paper
BPSO	Inertia weight, w	[0.9,0.4]
	Acceleration factors, c_1 and c_2	2
CCSA	Awareness probability, AP	0.1
	Flight length, fl	2

Table 4 represents the best fitness values of the feature selection algorithms. It can be seen that HLBPO shows better performance on 15 datasets and gives the best fitness values. Table 5 shows mean fitness values on 15 datasets. It can be inferred that HLBPO shows an optimal or near to optimal performance. The integration of hyper learning technique allows the algorithm to have excellent search capability. Table 6 shows that HLBPO gives optimal standard deviation on 9 datasets. Table 7 represents the feature selection ratio of this algorithm. If the ratio of feature selection is low, then the length of the optimal subset of selected features is also small. The results show us that HLBPO gained the optimal results in feature selection ratio in 20 datasets. HLBPO avoids local optima and identifies the best solution. Finally, the classification accuracy is compared with BDA, HLBDA, BPSO, and CCSA. The results are represented using the boxplot analysis in Figure 4. The boxplot is obtained after calculating the classification accuracy over all 21 datasets. HLBPO gives the best mean and median values. The results confirm the better performance of HLBPO in classification accuracy.

Table 4. Best Fitness Value results

Best Fitness Value											
No	**Dataset**	**HLBPO**	**HLBDA**	**BABC**	**BDA**	**BPSO**	**BMVO**	**CMAES**	**CCSA**	**LSHADE**	**BCOA**
1	Ionosphere	**0.0683**	0.0623	0.0830	0.0831	0.0914	0.0985	0.0746	0.0762	0.0719	0.0715
2	TOX_171	0.1688	**0.1300**	0.1897	0.1378	0.1522	0.1956	0.1959	0.1533	0.1383	0.1485
3	Colon	**0.0756**	0.0823	0.1154	0.0965	0.1131	0.1155	0.1135	0.1245	0.0992	0.0985
4	Leukemia	**0.0001**	0.0210	0.0457	0.0384	0.0256	0.0445	0.0457	0.0321	0.0313	0.0307
5	Hepatitis	**0.1130**	0.1154	0.1305	0.1245	0.1235	0.1226	0.1229	0.1825	0.1235	0.1220
6	Dermatology	**0.0111**	0.0129	0.0161	0.0132	0.0156	0.0215	0.0132	0.0274	0.0159	0.0161
7	ILPD	**0.2662**	0.2679	0.2722	0.3672	0.2972	0.2698	0.2672	0.2672	0.2672	0.2672
8	Lung discrete	**0.0431**	0.0553	0.0828	0.0597	0.0828	0.0846	0.0737	0.0718	0.0558	0.0690
9	Glass	**0.0067**	**0.0067**	**0.0067**	**0.0067**	**0.0067**	**0.0067**	**0.0067**	**0.0067**	**0.0067**	**0.0067**
10	Horse Colic	**0.1214**	0.1298	0.1350	0.1330	0.1309	0.1440	0.1298	0.1418	0.1327	0.1304
11	SPECT Heart	0.1509	0.1384	0.1415	0.1380	0.1361	0.1455	0.1388	0.1543	0.1396	**0.1337**
12	SCADI	**0.1139**	0.1154	0.1310	0.1190	0.1145	0.1312	0.1166	0.1320	0.1163	0.1158
13	Lymphography	**0.0143**	0.1116	0.1122	0.1181	0.1177	0.1297	0.1171	0.1309	0.1226	0.1257
14	Zoo	**0.0325**	0.0332	0.0332	**0.0325**	**0.0325**	0.0332	0.0334	0.0382	**0.0325**	**0.0325**
15	Arrhythmia	0.3003	**0.2936**	0.3330	0.3179	0.3282	0.3352	0.3269	0.3399	0.2999	0.3106
16	Soybean	**0.1124**	0.2009	0.2035	0.3073	0.2189	0.2421	0.2010	0.2294	0.2037	0.2035
17	LSVT	**0.1391**	0.2389	0.3003	0.2696	0.2621	0.2796	0.2866	0.4978	0.3010	0.3007
18	Musk 1	**0.0511**	0.0608	0.0879	0.0626	0.0783	0.0942	0.0739	0.0836	0.0633	0.0662
19	Primary Tumor	0.5985	0.5646	0.5673	0.5731	**0.5623**	0.5887	**0.5623**	0.5756	0.5883	0.5642
20	Seeds	0.0501	**0.0453**	**0.0453**	**0.0453**	**0.0453**	**0.0453**	**0.0453**	**0.0453**	**0.0453**	**0.0453**
21	Libras Movement	0.1837	**0.1665**	0.1939	0.1708	0.1913	0.2022	0.1751	0.1816	0.1688	0.1721

Table 5. Mean fitness value results

Mean Fitness Value											
No	**Dataset**	**HLBPO**	**HLBDA**	**BABC**	**BDA**	**BPSO**	**BMVO**	**CMAES**	**CCSA**	**LSHADE**	**BCOA**
1	Ionosphere	**0.0725**	0.0842	0.1016	0.0928	0.0958	0.1106	0.0882	0.1114	0.0909	0.0868
2	TOX_171	0.1930	**0.1575**	0.2137	0.1834	0.2070	0.2372	0.2194	0.2301	0.1780	0.1814
3	Colon	**0.0955**	0.1330	0.1626	0.1499	0.1508	0.1699	0.1605	0.1685	0.1436	0.1432
4	Leukemia	**0.0187**	0.0507	0.0694	0.0623	0.0606	0.0755	0.0684	0.0743	0.0626	0.0551
5	Hepatitis	**0.1245**	0.1311	0.1386	0.1368	0.1334	0.1454	0.1430	0.1457	0.1399	0.1425
6	Dermatology	0.0286	**0.0172**	0.0202	0.0192	0.0195	0.0254	0.0181	0.0238	0.0198	0.0209
7	ILPD	**0.2779**	0.2790	0.2804	0.2788	0.2792	0.2814	0.2845	0.2800	0.2837	0.2816
8	Lung discrete	**0.0657**	0.0774	0.0941	0.0835	0.0906	0.1039	0.0892	0.0979	0.0812	0.0834
9	Glass	**0.0111**	0.0112	**0.0111**	0.0112	**0.0111**	0.0116	0.0119	0.0116	0.0116	**0.0111**
10	Horse Colic	**0.1305**	0.1358	0.1480	0.1427	0.1409	0.1674	0.1419	0.1699	0.1479	0.1434
11	SPECT Heart	0.1662	**0.1507**	0.1572	0.1542	0.1539	0.1704	0.1644	0.1632	0.1598	0.1614
12	SCADI	**0.1235**	0.1259	0.1345	0.1310	0.1329	0.1413	0.1284	0.1410	0.1265	0.1291
13	Lymphography	**0.0143**	0.1311	0.1350	0.1359	0.1342	0.1527	0.1392	0.1515	0.1474	0.1416
14	Zoo	**0.0362**	0.0401	0.0397	0.0409	0.0369	0.0482	0.0470	0.0493	0.0503	0.0437
15	Arrhythmia	**0.3157**	0.3160	0.3450	0.3341	0.3413	0.3538	0.3352	0.3529	0.3270	0.3290
16	Soybean	**0.1240**	0.2124	0.2254	0.2212	0.2245	0.2594	0.2177	0.2479	0.2211	0.2168
17	LSVT	**0.1727**	0.3008	0.3172	0.3164	0.3175	0.3167	0.3206	0.3294	0.3338	0.3278
18	Musk 1	**0.0628**	0.0673	0.0961	0.0832	0.0929	0.1082	0.0842	0.1017	0.0793	0.0792
19	Primary Tumor	0.6127	0.5850	**0.5845**	0.5932	0.5850	0.6088	0.5935	0.5996	0.5989	0.5944
20	Seeds	0.0658	0.0557	**0.0529**	0.0557	**0.0529**	0.0532	0.0539	**0.0529**	0.0560	0.0532
21	Libras Movement	0.2030	**0.1813**	0.2027	0.1937	0.2006	0.2092	0.1890	0.2073	0.1899	0.1858

Table 6. Standard Deviation

Standard Deviation											
No	**Dataset**	**HLBPO**	**HLBDA**	**BABC**	**BDA**	**BPSO**	**BMVO**	**CMAES**	**CCSA**	**LSHADE**	**BCOA**
1	Ionosphere	**0.0049**	0.0086	0.0101	0.0105	0.0056	0.0053	0.0070	0.0090	0.0099	0.0104
2	TOX_171	0.0155	0.0213	0.0180	0.0273	0.0176	0.0156	**0.0140**	0.0199	0.0215	0.0182
3	Colon	**0.0150**	0.0335	0.0255	0.0349	0.0273	0.0307	0.0335	0.0266	0.0271	0.0298
4	Leukemia	**0.0091**	0.0133	0.0139	0.0152	0.0115	0.0154	0.0111	0.0144	0.0197	0.0133
5	Hepatitis	**0.0050**	0.0093	0.0057	0.0062	0.0080	0.0099	0.0132	0.0065	0.0091	0.0139
6	Dermatology	**0.0011**	0.0020	0.0022	0.0034	0.0019	0.0025	0.0022	0.0038	0.0035	0.0024
7	ILPD	0.0067	0.0051	0.0049	0.0049	**0.0046**	0.0054	0.0082	0.0051	0.0065	0.0064
8	Lung discrete	0.0120	0.0096	**0.0072**	0.0106	0.0088	0.0090	0.0077	0.0086	0.0104	0.0109
9	Glass	**0.0032**	0.0033	0.0033	0.0033	0.0033	0.0033	0.0036	**0.0032**	0.0033	0.0033
10	Horse Colic	**0.0030**	0.0039	0.0073	0.0088	0.0069	0.0164	0.0097	0.0140	0.0158	0.0110
11	SPECT Heart	0.0158	**0.0068**	0.0081	0.0082	0.0078	0.0098	0.0186	0.0098	0.0170	0.0205
12	SCADI	0.0065	0.0080	**0.0057**	0.0091	0.0062	0.0063	0.0074	0.0067	0.0079	0.0118
13	Lymphography	**0.0001**	0.0130	0.0125	0.0121	0.0138	0.0104	0.0151	0.0117	0.0147	0.0134
14	Zoo	0.0124	0.0079	0.0073	0.0080	**0.0065**	0.0101	0.0101	0.0097	0.0116	0.0092
15	Arrhythmia	0.0109	0.0094	**0.0042**	0.0088	0.0073	0.0077	0.0057	0.0065	0.0130	0.0100
16	Soybean	0.0079	0.0087	0.0089	0.0094	**0.0041**	0.0118	0.0116	0.0108	0.0109	0.0103
17	LSVT	0.0208	0.0312	**0.0190**	0.0273	0.0243	0.0210	0.0202	0.0220	0.0217	0.0226
18	Musk 1	0.0093	**0.0062**	0.0063	0.0099	0.0079	0.0076	0.0075	0.0079	0.0097	0.0077
19	Primary Tumor	0.0096	0.0104	0.0085	0.0099	0.0116	**0.0080**	0.0136	0.0120	0.0134	0.0135
20	Seeds	**0.0049**	0.0152	0.0057	0.0152	0.0057	0.0059	0.0066	0.0057	0.0152	0.0062
21	Libras Movement	0.0081	0.0084	0.0086	0.0104	0.0074	**0.0050**	0.0090	0.0092	0.0122	0.0102

Table 7. Feature selection ratio results

Feature Selection Ratio											
No	**Dataset**	**HLBPO**	**HLBDA**	**BABC**	**BDA**	**BPSO**	**BMVO**	**CMAES**	**CCSA**	**LSHADE**	**BCOA**
1	Ionosphere	**0.1000**	0.2191	0.2897	0.2676	0.2441	0.2882	0.2456	0.3426	0.2824	0.2265
2	TOX_171	**0.0761**	0.4794	0.5002	0.4829	0.4975	0.4451	0.4981	0.4982	0.4959	0.4591
3	Colon	**0.0325**	0.4378	0.4864	0.4629	0.4910	0.4444	0.4884	0.4884	0.4665	0.4178
4	Leukemia	**0.0216**	0.4517	0.4908	0.4695	0.4943	0.4180	0.4914	0.4937	0.4773	0.4144
5	Hepatitis	**0.1105**	0.3184	0.3158	0.3658	0.3263	0.3605	0.3500	0.3526	0.3368	0.3053
6	Dermatology	**0.3500**	0.4574	0.5574	0.4824	0.5338	0.5426	0.4926	0.5426	0.5074	0.4368
7	ILPD	**0.2700**	0.2950	0.3350	0.3150	0.3250	0.2800	0.3450	0.3200	0.2950	0.3050
8	Lung discrete	**0.1372**	0.3714	0.4860	0.4391	0.4808	0.4483	0.4703	0.4855	0.4378	0.3806
9	Glass	**0.2800**	0.2900	0.3050	0.2900	0.3050	0.3250	0.2900	0.3300	0.3050	0.3050
10	Horse Colic	**0.0703**	0.0870	0.2426	0.1370	0.1963	0.2574	0.1130	0.2926	0.1500	0.1056
11	SPECT Heart	**0.3500**	0.4477	0.4864	0.4500	0.5045	0.5273	0.4750	0.5250	0.4273	0.4159
12	SCADI	**0.0585**	0.2885	0.4373	0.3727	0.4249	0.4154	0.3980	0.4524	0.3488	0.3180
13	Lymphography	**0.1166**	0.4500	0.5083	0.4806	0.5000	0.4889	0.5083	0.4972	0.4333	0.4472
14	Zoo	0.4823	0.4563	0.4969	0.4469	**0.4250**	0.5188	0.4500	0.4938	0.4938	0.4531
15	Arrhythmia	**0.0562**	0.4050	0.4803	0.4699	0.4706	0.4303	0.4627	0.4787	0.4495	0.4048
16	Soybean	**0.0628**	0.6529	0.6429	0.6229	0.6414	0.5743	0.6286	0.5971	0.6486	0.6371
17	LSVT	**0.0151**	0.2844	0.4503	0.3482	0.4427	0.4006	0.4200	0.4494	0.3165	0.2984
18	Musk 1	**0.2398**	0.4687	0.4946	0.4783	0.4964	0.4602	0.4946	0.5033	0.4849	0.4458
19	Primary Tumor	**0.4823**	0.6676	0.6706	0.6118	0.6676	0.5941	0.6265	0.6441	0.6412	0.6059
20	Seeds	**0.3000**	0.3143	0.3143	0.3143	0.3143	0.3214	0.3429	0.3143	0.3214	0.3214
21	Libras Movement	**0.1888**	0.4161	0.4600	0.4517	0.4489	0.4311	0.4300	0.4644	0.4322	0.4061

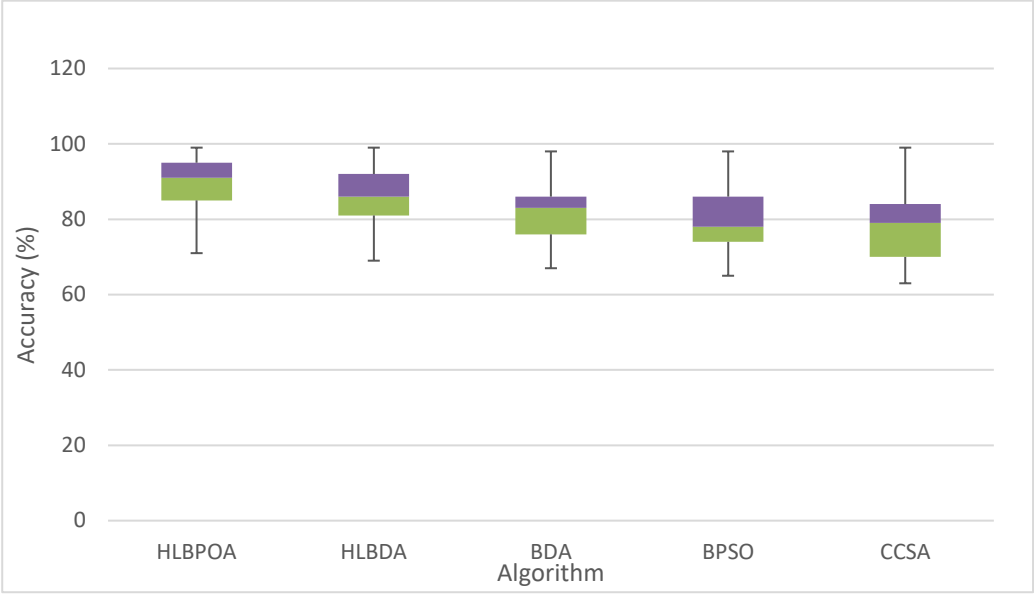

Fig. 4. Boxplots Analysis of HLBPO

5.4 Analysis of Results:

Based on the findings, it is clear that HLBPO has an outstanding performance when it comes to finding a solution to the issue of feature selection. The most optimal answer was derived by the HLBPO from the extensive list of characteristics. For all of the iterations, the HLBPO showed a strong acceleration rate, which was important for the convergence process. The findings shown in Tables 5, 6, and 7 demonstrate that the HLBPO method has a higher rate of convergence than any of the other feature selection algorithms with which it is contrasted.

The performance of the HLBPO feature selection algorithm as a means of resolving the feature selection issue is the subject of discussion in this work. According to the findings, HLBPO performed better than other feature selection algorithms, successfully identifying the optimal solution from among a diverse group of characteristics. In addition, HLBPO showed a decent acceleration rate all the way through the iterations, and it also revealed a superior convergence rate than the other algorithms. In addition, the HLBPO produced the best classification results by picking high-quality features, which resulted in an increase in the accuracy of the prediction. With high-dimensional datasets like Colon and Leukemia, the algorithm was also effective in selecting the characteristics that were most relevant to the problem at hand and excluding those that were unnecessary.

The accomplishments of HLBPO may largely be credited to two key contributors. The first thing that HLBPO accomplished was add the idea of the global best solution. This made it possible for candidates to gain knowledge from it and break free of local optimal solutions if the personal or global best solution was being held captive by those solutions. Second, HLBPO used a brand new approach to improve the position of candidate solutions, which made it possible for the company to progress towards both personal and worldwide best solutions. Because of this method, the searching behavior of the algorithm is improved, which enables it to converge more quickly and locate the global optima. When compared to other well-known feature selection algorithms, the findings reveal that HLBPO is an effective feature selection method that achieves better performance.

The proposed Hyper Learning-Based Binary Political Optimizer (HLBPO) algorithm, while innovative and promising, may have certain limitations that should be considered. Like many metaheuristic algorithms, the performance of HLBPO may be sensitive to the selection of its hyperparameters. Tuning these parameters effectively to achieve optimal results across different datasets and problem domains can be challenging. As a metaheuristic algorithm, HLBPO may require significant computational resources, especially when dealing with large-scale datasets or high-dimensional feature spaces. The algorithm's complexity may limit its applicability in real-time or resource-constrained environments. The effectiveness of HLBPO may vary across different types of datasets and problem instances. The algorithm's performance may not generalize well to diverse and complex feature selection tasks, leading to suboptimal results in certain scenarios.

5.5 Conclusion and Future Work:

In this study, the relevance of feature selection as a preprocessing activity to increase the accuracy of classification by machine learning algorithms is discussed. The difficulty of feature selection is addressed by the introduction of a newly designed algorithm called HLBPO. The performance of the Political Optimizer algorithm is improved with the help of a hyper-learning method that is used by the algorithm. Using twenty-one different classification data sets, the performance of the method was analysed and compared to nine other optimization techniques that are considered to be state-of-the-art. According to the findings, HLBPO performed better than the majority of the algorithms that were evaluated, both in terms of decreasing the total amount of features and boosting the accuracy of classification.

The low feature selection ratio of HLBPO gives the impression that the algorithm chooses just the most essential characteristics from the extensive pool of features that are accessible. This not only decreases the total number of

characteristics, but it also increases the accuracy of the classification. The findings of the research indicate that the HLBPO algorithm is an efficient one for feature selection.

The Hyper Learning-Based Binary Political Optimizer (HLBPO) algorithm has the potential to significantly improve feature selection in machine learning. The algorithm's demonstrated effectiveness in reducing feature dimensionality and improving classification accuracy contributes to its practical relevance in domains such as healthcare, finance, and engineering. HLBPO's low feature selection ratio indicates that it selects only the most vital features, leading to reduced processing time and improved efficiency. The algorithm's potential for generalizability across diverse problem domains, its innovative approach to hyper-learning, and its comparative analysis with state-of-the-art feature selection algorithms contribute to its impact and relevance in the field of metaheuristic algorithms. The study's findings encourage further research into the algorithm's applicability to specific domains, its scalability to big data scenarios, and its robustness under noisy or uncertain conditions. Overall, the impact analysis positions HLBPO as a valuable contribution to the field of feature selection and machine learning, with promising practical and research-oriented impacts.

The HLBPO method has the potential to be used in a variety of different applications in the not-too-distant future, including the identification of COVID-19 from pictures, electromyography, power quality diagnostics, and optimized deep neural networks. In addition, the readers are encouraged to study other initialization procedures with the purpose of further enhancing the performance of the algorithm. The findings of the research indicate that HLBPO has the potential to function as a feature selection algorithm that is both effective and efficient.

References

1. X.-S. Yang, "Nature-inspired optimization algorithms: challenges and open problems," Journal of Computational Science, p. 101104, 2020. [https://doi.org/10.1016/j.jocs.2020.101104]
2. S.-C. Wang, "Genetic algorithm," in Interdisciplinary Computing in Java Programming, Springer, 2003, p. 101–116.
3. P. Merz, B. Freisleben and others, "Fitness landscapes and memetic algorithm design," New ideas in optimization, p. 245–260, 1999.
4. K. E. Kinnear, W. B. Langdon, L. Spector, P. J. Angeline and U.-M. O'Reilly, Advances in genetic programming, vol. 3, MIT press, 1994.
5. E. Alba and J. M. Troya, "Cellular evolutionary algorithms: Evaluating the influence of ratio," in International Conference on Parallel Problem Solving from Nature, 2000. [https://doi.org/10.1007/3-540-45356-3_3]

6. M. Dorigo, M. Birattari and T. Stutzle, "Ant colony optimization," IEEE computational intelligence magazine, vol. 1, p. 28–39, 2006. [10.1109/MCI.2006.329691]
7. S. Mirjalili and A. Lewis, "The whale optimization algorithm," Advances in engineering software, vol. 95, p. 51–67, 2016. [https://doi.org/10.1016/j.advengsoft.2016.01.008]
8. E. Emary, H. M. Zawbaa and A. E. Hassanien, "Binary ant lion approaches for feature selection," Neurocomputing, vol. 213, p. 54–65, 2016. [http://dx.doi.org/10.1016/j.advengsoft.2016.01.008]
9. S. Mirjalili, "Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems," Neural Computing and Applications, vol. 27, p. 1053–1073, 2016. [https://doi.org/10.1007/s00521-015-1920-1]
10. D. Karaboga and B. Akay, "A comparative study of artificial bee colony algorithm," Applied mathematics and computation, vol. 214, p. 108–132, 2009. [10.1016/j.amc.2009.03.090]
11. R. A. Formato, "Improved CFO algorithm for antenna optimization," Progress in Electromagnetics Research, vol. 19, p. 405–425, 2010.
12. T. Grinshpoun and A. Meisels, "Completeness and performance of the APO algorithm," Journal of Artificial Intelligence Research, vol. 33, p. 223–258, 2008. [https://doi.org/10.1613/jair.2611]
13. E. Rashedi, H. Nezamabadi-Pour and S. Saryazdi, "GSA: a gravitational search algorithm," Information sciences, vol. 179, p. 2232–2248, 2009. [10.1016/j.ins.2009.03.004]
14. A. H. Kashan, "League Championship Algorithm (LCA): An algorithm for global optimization inspired by sport championships," Applied Soft Computing, vol. 16, p. 171–200, 2014. [https://doi.org/10.1016/j.asoc.2013.12.005]
15. A. Ahmadi-Javid, "Anarchic Society Optimization: A human-inspired method," in 2011 IEEE Congress of Evolutionary Computation (CEC), 2011. [https://doi.org/10.1109/CEC.2011.5949940]
16. M. O'Neill and C. Ryan, "Grammatical evolution," IEEE Transactions on Evolutionary Computation, vol. 5, p. 349–358, 2001. [https://doi.org/10.1109/4235.942529]
17. Q. Askari, I. Younas and M. Saeed, "Political Optimizer: A novel socio-inspired meta-heuristic for global optimization," Knowledge-Based Systems, p. 105709, 2020. [https://doi.org/10.1016/j.knosys.2020.105709]
18. G. Manita and O. Korbaa, "Binary Political Optimizer for Feature Selection Using Gene Expression Data," Computational Intelligence and Neuroscience, vol. 2020, 2020. [https://doi.org/10.1155/2020/8896570]
19. H. Dong, J. Sun, X. Sun and R. Ding, "A many-objective feature selection for multi-label classification," Knowledge-Based Systems, vol. 208, p. 106456, 2020. [https://doi.org/10.1016/j.knosys.2020.106456]
20. N. Al-Madi, H. Faris and S. Mirjalili, "Binary multi-verse optimization algorithm for global optimization and discrete problems," International Journal of Machine Learning and Cybernetics, vol. 10, p. 3445–3465, 2019. [https://doi.org/10.1007/s13042-019-00931-8]

21. M. M. Mafarja and S. Mirjalili, "Hybrid whale optimization algorithm with simulated annealing for feature selection," Neurocomputing, vol. 260, p. 302–312, 2017. [https://doi.org/10.1016/j.neucom.2017.04.053]
22. M. Abdel-Basset, D. El-Shahat, I. El-henawy, V. H. C. de Albuquerque and S. Mirjalili, "A new fusion of grey wolf optimizer algorithm with a two-phase mutation for feature selection," Expert Systems with Applications, vol. 139, p. 112824, 2020. [https://doi.org/10.1016/j.eswa.2019.112824]
23. M. Mafarja and S. Mirjalili, "Whale optimization approaches for wrapper feature selection," Applied Soft Computing, vol. 62, p. 441–453, 2018. [https://doi.org/10.1016/j.asoc.2017.11.006]
24. L. M. Abualigah, A. T. Khader and E. S. Hanandeh, "A new feature selection method to improve the document clustering using particle swarm optimization algorithm," Journal of Computational Science, vol. 25, p. 456–466, 2018. [https://doi.org/10.1016/j.jocs.2017.07.018]
25. M. Ghosh, R. Guha, R. Sarkar and A. Abraham, "A wrapper-filter feature selection technique based on ant colony optimization," Neural Computing and Applications, p. 1–19, 2019. [https://doi.org/10.1007/s00521-019-04171-3]
26. M. H. Aghdam, P. Kabiri and others, "Feature selection for intrusion detection system using ant colony optimization.," IJ Network Security, vol. 18, p. 420–432, 2016.
27. A. M. Anter, A. E. Hassanien, M. A. ElSoud and T.-H. Kim, "Feature selection approach based on social spider algorithm: case study on abdominal CT liver tumor," in 2015 Seventh International Conference on Advanced Communication and Networking (ACN), 2015. [https://doi.org/10.1109/ACN.2015.32]
28. S. Taghian, M. H. Nadimi-Shahraki and H. Zamani, "Comparative analysis of transfer function-based binary Metaheuristic algorithms for feature selection," in 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), 2018. [https://doi.org/10.1109/IDAP.2018.8620828]
29. S. Sabba and S. Chikhi, "A discrete binary version of bat algorithm for multidimensional knapsack problem," International Journal of Bio-Inspired Computation, vol. 6, p. 140–152, 2014. [https://doi.org/10.1504/IJBIC.2014.060598]
30. X. Jun and H. Chang, "The discrete binary version of the improved particle swarm optimization algorithm," in 2009 International Conference on Management and Service Science, 2009. [https://doi.org/10.1109/ICMSS.2009.5302726]
31. A. G. Hussien, D. Oliva, E. H. Houssein, A. A. Juan and X. Yu, "Binary Whale Optimization Algorithm for Dimensionality Reduction," Mathematics, vol. 8, p. 1821, 2020. [https://doi.org/10.3390/math8101821]
32. M. A. Khanesar, M. Teshnehlab and M. A. Shoorehdeli, "A novel binary particle swarm optimization," in 2007 Mediterranean conference on control & automation, 2007. [https://doi.org/10.1109/MED.2007.4433821]
33. M. M. Mafarja, D. Eleyan, I. Jaber, A. Hammouri and S. Mirjalili, "Binary dragonfly algorithm for feature selection," in 2017 international conference on new trends in computing sciences (ICTCS), 2017. [https://doi.org/10.1109/ICTCS.2017.43]

34. A. G. Hussien, A. E. Hassanien, E. H. Houssein, S. Bhattacharyya and M. Amin, "S-shaped binary whale optimization algorithm for feature selection," in Recent trends in signal and image processing, Springer, 2019, p. 79–87. [https://doi.org/10.1007/978-981-10-8863-6_9]
35. A. G. Hussien, E. H. Houssein and A. E. Hassanien, "A binary whale optimization algorithm with hyperbolic tangent fitness function for feature selection," in 2017 Eighth international conference on intelligent computing and information systems (ICICIS), 2017. [https://doi.org/10.1109/INTELCIS.2017.8260031]
36. H. Faris, M. M. Mafarja, A. A. Heidari, I. Aljarah, A.-Z. Ala'M, S. Mirjalili and H. Fujita, "An efficient binary salp swarm algorithm with crossover scheme for feature selection problems," Knowledge-Based Systems, vol. 154, p. 43–67, 2018. [https://doi.org/10.1016/j.knosys.2018.05.009]
37. K. Taunk, S. De, S. Verma and A. Swetapadma, "A Brief Review of Nearest Neighbor Algorithm for Learning and Classification," in 2019 International Conference on Intelligent Computing and Control Systems (ICCS), 2019. [https://doi.org/10.1109/ICCS45141.2019.9065747s]
38. M. Lichman, UCI Machine Learning Repository [https://archive. ics. uci. edu/ml/datasets. html]. Irvine, CA: University of California, School of Information and Computer Science, Accessed, 2016.
39. Datasets | Feature Selection @ ASU.
40. J. Too and S. Mirjalili, "A hyper learning binary dragonfly algorithm for feature selection: A COVID-19 case study," Knowledge-Based Systems, vol. 212, p. 106553, 2021. [https://doi.org/10.1016/j.knosys.2020.106553]
41. Y. He, H. Xie, T.-L. Wong and X. Wang, "A novel binary artificial bee colony algorithm for the set-union knapsack problem," Future Generation Computer Systems, vol. 78, p. 77–86, 2018. [https://doi.org/10.1016/j.future.2017.05.044]
42. J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in 1997 IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation, 1997. [https://doi.org/10.1109/ICSMC.1997.637339]
43. N. Hansen and S. Kern, "Evaluating the CMA evolution strategy on multimodal test functions," in International Conference on Parallel Problem Solving from Nature, 2004. [https://doi.org/10.1007/978-3-540-30217-9_29]
44. G. I. Sayed, A. E. Hassanien and A. T. Azar, "Feature selection via a novel chaotic crow search algorithm," Neural computing and applications, vol. 31, p. 171–188, 2019. [https://doi.org/10.1007/s00521-017-2988-6]
45. R. Tanabe and A. S. Fukunaga, "Improving the search performance of SHADE using linear population size reduction," in 2014 IEEE congress on evolutionary computation (CEC), 2014. [https://doi.org/10.1109/CEC.2014.6900380]
46. R. C. T. de Souza, C. A. de Macedo, L. dos Santos Coelho, J. Pierezan and V. C. Mariani, "Binary coyote optimization algorithm for feature selection," Pattern Recognition, vol. 107, p. 107470, 2020. [https://doi.org/10.1016/j.patcog.2020.107470]
47. Q. Askari, I. Younas and M. Saeed, "Political Optimizer: A novel socio-inspired meta-heuristic for global optimization," Knowledge-Based Systems, p. 105709, 3 2020. [https://doi.org/10.1016/j.knosys.2020.105709]

48. Hancer, Emrah. "An improved evolutionary wrapper-filter feature selection approach with a new initialisation scheme." Machine Learning (2021): 1-24. [https://doi.org/10.1007/s10994-021-05990-z]

49. Tsamardinos, Ioannis, et al. "A greedy feature selection algorithm for Big Data of high dimensionality." Machine learning 108.2 (2019): 149-202. [https://doi.org/10.1007/s10994-018-5748-7]

50. Shang, Ronghua, et al. "Unsupervised feature selection based on kernel fisher discriminant analysis and regression learning." Machine Learning 108.4 (2019): 659-686. [https://doi.org/10.1007/s10994-018-5765-6]

51. Singha, Sumanta, and Prakash P. Shenoy. "An adaptive heuristic for feature selection based on complementarity." Machine Learning 107.12 (2018): 2027-20 [https://doi.org/10.1007/s10994-018-5728-y]

52. Bacanin, Nebojsa, et al. "Addressing feature selection and extreme learning machine tuning by diversity-oriented social network search: an application for phishing websites detection." Complex & Intelligent Systems 9.6 (2023): 7269-7304. [https://link.springer.com/article/10.1007/s40747-023-01118-z]

53. Ganesh, Narayanan, et al. "Efficient feature selection using weighted superposition attraction optimization algorithm." Applied Sciences 13.5 (2023): 3223. [https://www.mdpi.com/2076-3417/13/5/3223]

54. Bacanin, Nebojsa, et al. "A novel firefly algorithm approach for efficient feature selection with COVID-19 dataset." Microprocessors and Microsystems 98 (2023): 104778. [https://www.sciencedirect.com/science/article/pii/S0141933123000248]

55. Kareem, Saif S., et al. "An effective feature selection model using hybrid metaheuristic algorithms for iot intrusion detection." Sensors 22.4 (2022): 1396. [https://www.mdpi.com/1424-8220/22/4/1396]

56. Zivkovic, Miodrag, et al. "Novel improved salp swarm algorithm: An application for feature selection." Sensors 22.5 (2022): 1711. [https://www.mdpi.com/1424-8220/22/5/1711]

57. Akinola, Olatunji A., et al. "A hybrid binary dwarf mongoose optimization algorithm with simulated annealing for feature selection on high dimensional multi-class datasets." Scientific Reports 12.1 (2022): 14945. [https://www.nature.com/articles/s41598-022-18993-0]

58. Askari, Qamar, and Irfan Younas. "Political optimizer based feedforward neural network for classification and function approximation." Neural Processing Letters 53.1 (2021): 429-458. [https://doi.org/10.1007/s11063-020-10406-5]

59. Pashaei, Elham. "Mutation-based Binary Aquila optimizer for gene selection in cancer classification." Computational Biology and Chemistry 101 (2022): 107767. [https://doi.org/10.1016/j.compbiolchem.2022.107767]

60. Pashaei, Elnaz, and Elham Pashaei. "Hybrid binary COOT algorithm with simulated annealing for feature selection in high-dimensional microarray data." Neural Computing and Applications 35.1 (2023): 353-374. [https://doi.org/10.1007/s00521-022-07780-7]