Feature Selection using Hyper Learning-Based Binary Political Optimizer Algorithm

Mehroze Khan, Irfan Younas, Maryam Bashir*

*Corresponding Author Email:maryam.bashir@nu.edu.pk

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

Abstract.

Over the past decade, there has been an exponential surge in the volume of data and information. This escalation has given rise to the challenge known as the "curse of dimensionality," where an abundance of features can detrimentally impact the performance of machine learning algorithms. Feature selection, aimed at identifying a subset of features to enhance machine learning model performance, is pivotal. However, feature selection poses a significant challenge as it entails an optimization problem with an exponential search space, rendering it NP-hard. Over the last decade, metaheuristics have proven successful in addressing NP-hard problems. In this article, we introduce a novel algorithm, the Hyper Learning-based Binary Political Optimizer (HLBPO), designed to pinpoint the optimal feature subset. HLBPO represents an advancement over the Political Optimizer Algorithm, incorporating a hyper learning strategy that enables the algorithm to navigate away from local minima solutions, thereby improving its search behavior. To assess the efficacy of the proposed algorithm, various evaluation metrics are employed. A comparative analysis is conducted against nine state-of-the-art algorithms, revealing the superior performance of HLBPO. Specifically, HLBPO demonstrates its effectiveness in reducing the number of features while concurrently enhancing classification accuracy.

Keywords: Feature Selection, Political Optimizer, Hyper Learning, Optimization, Classification.

1 Introduction

The exponential growth of data and information has spurred innovation in advanced data pre-processing techniques among data mining and data science researchers. A significant challenge arising from this surge is the presence of a large number of dimensions or features in the data. Many of these features are characterized by noise, adversely impacting the performance of machine learning algorithms. Feature selection emerges as a crucial process in addressing this issue, aiming to reduce the data's dimensions to enhance the time complexity and prediction accuracy of machine learning algorithms. By selectively choosing

important features and discarding others, the data size is reduced, resulting in improved processing time and algorithm accuracy. This step holds paramount importance in the domains of data mining, machine learning, and statistics, finding applications in diverse fields such as medicine, bioinformatics, social media, multimedia retrieval, and more.

Over the past decade, the literature has seen the emergence of numerous feature selection techniques. The process involves two key steps: feature subset selection and the evaluation of the selected subset of features. The feature subset is chosen using a specific search strategy, and its efficacy is assessed using a machine learning classifier. Feature selection techniques can be broadly categorized into two groups: filter-based methods and wrapper-based methods. Filter-based methods leverage data-specific characteristics to evaluate selected features, while wrapper-based methods utilize the prediction accuracy of a classifier for evaluating the chosen feature subset. Although filter-based methods are more efficient in terms of time complexity, wrapper-based methods outperform them in classifier accuracy.

The task of selecting the optimal subset of features presents itself as an optimization problem. As the number of dimensions increases, so does the size of the search space, often reaching an exponential magnitude. This exponential growth renders the problem NP-hard when employing exact algorithms. To tackle such challenges, approximate algorithms come into play, providing solutions close to optimal. Over the past decade, metaheuristics have demonstrated remarkable efficacy in addressing NP-hard problems. Metaheuristics serve as a framework for algorithm development, offering fundamental guidelines.

Nature-inspired metaheuristics, in particular, have proven to be adaptable and effective for solving optimization problems. These algorithms draw inspiration from natural processes, leveraging this insight to address real-world problems. Examples of evolution-inspired algorithms include Genetic Algorithm, Memetic Algorithm, Genetic Programming, and Cellular Evolutionary Algorithm. Swarm intelligence, focusing on collective behavior in living organisms, incorporates algorithms such as Ant Colony Optimization, Whale Optimization Algorithm, Grey Wolf Optimization Algorithm, Dragonfly Algorithm, and Bee Colony Algorithm. Physics-inspired algorithms, such as the Carrier Frequency Offset (CFO) Algorithm, Asynchronous Partial Overlay (APO) Algorithm, and Gravitational Search Algorithm (GSA), take inspiration from physical phenomena. Additionally, Human Behavior-inspired algorithms, including the League Championship Algorithm, Anarchic Society Optimization, and Grammatical Evolution Algorithm, have gained prominence.

In recent years, researchers have increasingly applied nature-inspired metaheuristics to the challenge of feature selection, attracted by their outstanding performance. For instance, Basset et al. introduced a Grey Wolf Optimizer algorithm with a two-phase mutation for feature selection, while Hussien et al. proposed a Binary Whale Optimization Algorithm for dimensionality reduction. The following section delves into more algorithms as part of the related work.

The Political Optimizer (PO) [17] is a recently introduced socio-inspired algorithm designed for solving optimization problems, drawing inspiration from human behavior in politics. The algorithm replicates the dynamic part-switching behavior observed in political candidates. It mirrors key phases of an electoral process, such as the formation of political parties, assignment of constituencies to candidates, selection of party leaders and constituency winners, party-switching, and parliamentary affairs. Noteworthy features of this algorithm include the dual role each candidate solution assumes, acting as a member of a political party and a participant in an election. This dual role facilitates two updates for every solution, enhancing the chances of improvement. The algorithm also incorporates a novel strategy for updating candidate positions based on learned political behaviors. PO demonstrated superior performance in comparison to 15 other algorithms, showcasing commendable convergence speed and exploration.

To extend its applicability to discrete problems, a Binary Political Optimizer (BPO) [18] was later proposed. While BPO exhibited strong convergence and accuracy across nine datasets compared to eight algorithms, it exhibited a tendency to get stuck in local optima due to its robust exploitation. To address this limitation, the Hyper Learning-Based Binary Political Optimizer (HLBPO) algorithm is introduced. This algorithm is specifically designed to tackle feature selection problems, aiming to identify the optimal subset of features to enhance the classification accuracy of machine learning algorithms. HLBPO demonstrated outstanding performance when compared to nine state-of-the-art algorithms for feature selection.

The main contributions of this study are summarized as follows:

1. Introduction of the novel HLBPO algorithm for feature selection problems.
2. Enhancement of the exploratory rate of the binary PO through a hyper learning strategy, improving its exploratory behavior.
3. Comparative analysis of HLBPO performance with nine state-of-the-art algorithms across twenty-one datasets of varying dimensions.
4. Demonstration of HLBPO's superiority over nine state-of-the-art algorithms for feature selection.

The subsequent sections of the article are organized as follows: Section 2 provides a literature review of socio-inspired algorithms related to feature selection problems. Section 3 offers an overview of the Political Optimizer Algorithm. Section 4 introduces the proposed Hyper Learning-Based Binary Political Optimizer algorithm (HLBPO). Section 5 details experiments, results, and comparisons with other feature selection algorithms. Finally, Section 6 concludes the paper and outlines directions for future work.

2 Related Work

In recent years, numerous metaheuristics have been introduced to address feature selection problems. This section provides an overview of recent work related to feature selection using metaheuristics.

To enhance the effectiveness of feature selection, hybrid algorithms have gained popularity. Hybrid algorithms offer the advantage of leveraging the strengths of two distinct algorithms. One such hybrid binary optimization algorithm [20] was proposed, combining the particle swarm algorithm with the binary grey wolf algorithm. This hybrid approach demonstrated enhanced performance, coupled with a reduction in the feature selection ratio. Utilizing the wrapper method of k-nearest neighbor, optimal solutions were sought and benchmarked across 18 datasets. Another noteworthy hybrid algorithm [21] employed a unique technique for feature selection, integrating a simulated annealing algorithm with the whale optimization algorithm. Initially, the whale optimization algorithm explored the most promising regions, and subsequently, the simulated annealing algorithm increased the exploitation of these regions. This hybrid algorithm, evaluated against 18 datasets, exhibited superior classification accuracy compared to other wrapper-based methods.

A two-phase mutation grey wolf optimization algorithm [22] has been proposed to address the feature selection problem. In the first mutation phase, the algorithm reduces features while maintaining classification accuracy, and in the second phase, it introduces important features to enhance classification. Leveraging the k-nearest neighbor wrapper method, the algorithm is benchmarked against 35 datasets, demonstrating improved performance.

A binary whale optimization algorithm [23] introduces two binary invariants. The first invariant utilizes roulette and tournament selection instead of a random operator, while the second invariant employs crossover and mutation operators for exploitation. An ant colony algorithm [25] combines the filter method and the wrapper method. It evaluates the subset using the filter method, reducing computational complexity, and includes a memory to retain the best ants. The MMFS algorithm [19] is tailored for multi-label classification problems. It enhances the convergence rate and diversity of the NSGA-III many-objective algorithm by introducing two archives. The algorithm employs uniform crossover and mutation operators to improve exploration. Tested on 11 datasets with multiple labels, MMFS exhibits good classification results, eliminates unnecessary features, and balances multiple objectives, albeit with a substantial cost for calculating wrapper features. An improved differential evolution algorithm [48] is proposed for feature selection using the wrapper-filter feature subset selection approach. Fuzzy estimators initialize solutions with a predefined number of best features, aiding in identifying optimal features from the global search space. Another differential evolution-based algorithm for feature selection is the Success History-based Adaptive Differential Evolution Algorithm with Linear Population Size Reduction (LSHADE) [45]. The Chaotic Crow Search Algorithm

(CCSA) [44] is introduced for feature selection and undergoes testing on 20 benchmark datasets. CCSA outperforms various existing algorithms for feature selection.

Recognizing feature selection as a binary optimization problem, recent studies have transformed continuous-valued algorithms into binary counterparts [28] to reduce space and computational complexities. The binary Bat Algorithm [29] is introduced with a focus on minimizing space requirements, utilizing an s-shaped transfer function to convert continuous values into binary ones. Binary algorithms can also be integrated with other algorithms; the BSAPSO algorithm [30], combining a binary particle swarm-based approach with simulated annealing, demonstrates improved convergence speed. Another study [31] presents two binary versions of the whale optimization algorithm, utilizing s-shaped and v-shaped transfer functions, proving more effective than other binary socio-inspired algorithms.

Several binary versions of evolutionary algorithms for feature selection have been developed, including the hyper-learning binary dragonfly algorithm (HLBDA) [40], binary artificial bee colony (BABC) [41], binary dragonfly algorithm (BDA) [33], binary coyote optimization algorithm (BCOA) [46], binary particle swarm optimization (BPSO) [42], and binary multiverse optimizer (BMVO) [20].

Despite numerous studies on evolutionary algorithms for feature selection, there remains room for improvement. This study introduces a hyper-learning-based binary political optimizer for feature selection, with details of the political optimizer algorithm [17] provided in the subsequent section.

3 Political Optimizer

The Political Optimizer (PO) Algorithm, proposed in 2020 [17], is a socio-inspired approach designed to address the feature selection problem. Drawing inspiration from the political system within our social environment, this algorithm integrates various phases of politics. Exploration and exploitation are facilitated through the utilization of a recent past-based position updating strategy (RPPUS), enabling the algorithm to learn from prior elections. This learning process involves enhancing underperforming solutions to achieve better results, akin to party leaders, and adjusting positions by comparing them with constituency winners. The equilibrium between exploration and exploitation is achieved through the party-switching phase. Fitness evaluation of candidate solutions is carried out during the election phase. To introduce elements of exploitation and convergence, the algorithm incorporates a parliamentary phase. The flowchart of the PO algorithm is depicted in Figure 1.

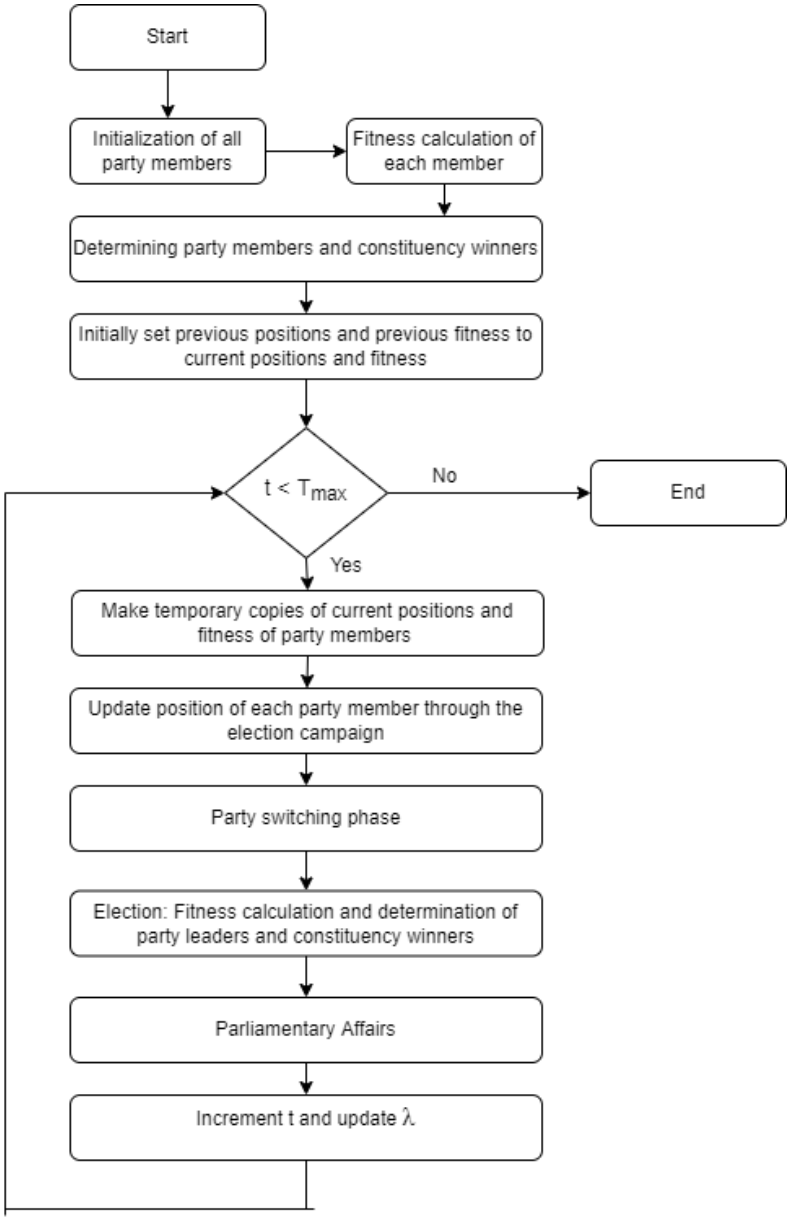

Fig. 1. Flowchart of PO Algorithm [17]

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. Each party P_i has n members in it. The jth member of an ith party is represented as p_i^j. The members are represented by using a d-dimensional vector which is the number of input variables. The kth dimension is represented as $p_{i,k}^j$ which means jth member of ith political party in the kth dimension.

The constituencies are divided into n different constituencies. The j^{th} member of each party competes for an election in the j^{th} constituency. The set of selected party leaders and constituency winners are represented and these sets are calculated in the election phase.

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 the RPPUS strategy. 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.

3.3 Party Switching

This phase is used to balance the exploration as well as exploitation. There is a parameter named party-switching rate represented as λ that starts from the value λmax and then decreases to zero. All the members are selected with the probability of λ and then switched with the least fit member of a randomly selected party.

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.

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.

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.

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 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

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

Algorithm 3: Election (n)

foreach Pi ∈ P do

foreach $p_i^j \in$ 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).

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

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

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)
        r1 ← random number in the interval [0, 1]
        if 0 ≤ r1 < plr then
            Update position using Eq. (14)
        elseif plr ≤ r1 < glr then
            Update position with respect to personal best solution
        elseif glr ≤ r1 ≤ 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].

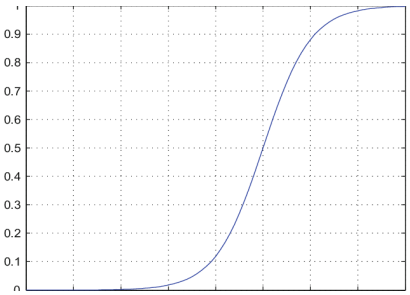

Fig. 2. S-shaped Transfer Function [34]

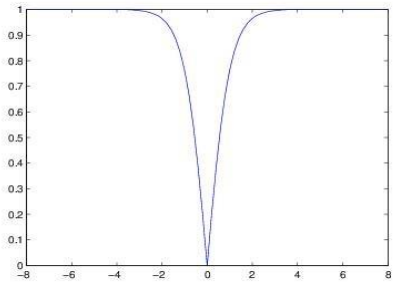

Fig. 3. V-shaped Transfer Function [35]

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. (18).

$$Fitness = \alpha ER + \beta \left(\frac{|SF|}{|TF|}\right) \qquad (18)$$

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 Acceleration factors, c_1 and c_2	[0.9,0.4] 2
CCSA	Awareness probability, AP Flight length, fl	0.1 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. 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

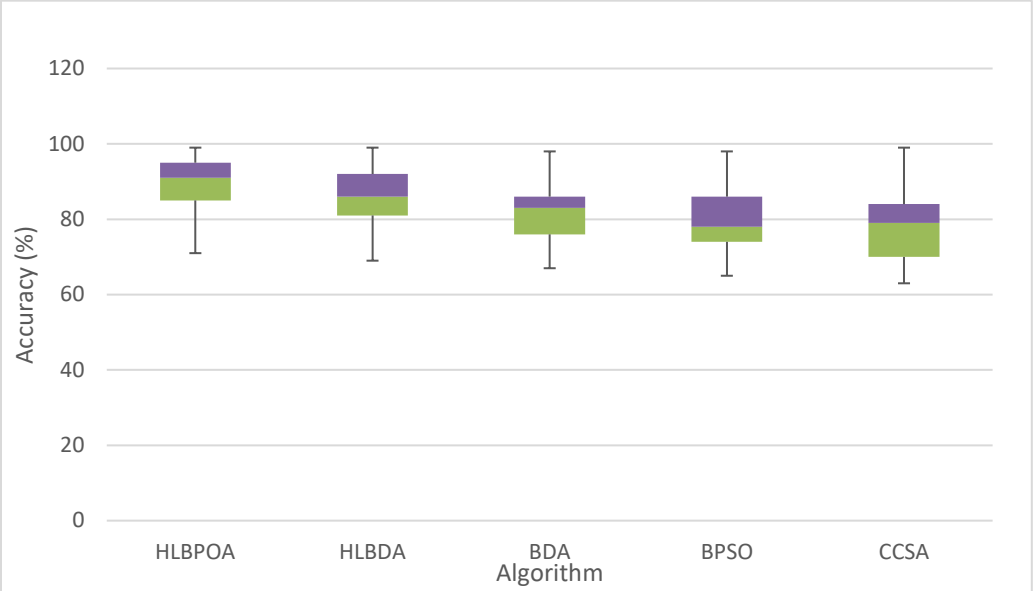

Fig. 3. Boxplots Analysis of HLBPO

5.4 Analysis of Results:

From the results, it can be seen that HLBPO has an excellent performance in solving the feature selection problem. The HLBPO reached the best solution from the complex set of features. In terms of convergence, HLBPO had a good acceleration rate throughout the iterations. The results in Tables 5, 6, and 7 show that HLBPO has a better convergence rate than any other feature selection algorithm it is compared to.

It is also seen that HLBPO has produced the best classification results. The features that are being selected by this algorithm are top quality, which improves the prediction accuracy. In datasets with very high dimensions like Colon and Leukemia, this algorithm chose the most relevant features discarding all the redundant features.

This improvement in results is possible because of two main factors. Firstly, HLBPO incorporated the concept of the global best solution rather than only moving towards the personal best solution (party leader) which enables the candidates to learn from their global best solutions. This allows the algorithm to escape from the local optima solutions if the personal or global best solution is trapped in it. Secondly, HLBPO introduced a new strategy to

update the position of candidate solutions which improved the searching behavior. This strategy enabled the candidate solutions to move towards personal best and global best solutions. This allows the HLBPO to converge faster and find the global optima.

6 Conclusion and future work

The process of feature selection is crucial for enhancing the classification accuracy of machine learning algorithms as part of preprocessing tasks. This study introduces a novel algorithm, the Hyper Learning-Based Political Optimizer (HLBPO), designed to address the feature selection problem. Employing a hyper-learning strategy, the HLBPO algorithm builds upon the foundation of the Political Optimizer algorithm to improve its overall performance. Through evaluations on twenty-one classification datasets, a comparative analysis with nine state-of-the-art optimization algorithms showcases the remarkable superiority of the proposed algorithm. HLBPO exhibits a minimal feature selection ratio, indicating its ability to discern and utilize a critical set of features from a complex array. Notably, HLBPO not only reduces the number of features but also enhances classification accuracy. This study concludes that HLBPO proves to be an effective algorithm for feature selection tasks.

Looking ahead, the HLBPO algorithm holds promise for applications in diverse domains, including images, electromyography, power quality diagnosis, and optimized deep neural networks. Future research avenues may explore additional initialization strategies to further enhance the algorithm's performance.

References

1. X.-S. Yang, "Nature-inspired optimization algorithms: challenges and open problems," Journal of Computational Science, p. 101104, 2020.
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.

6. M. Dorigo, M. Birattari and T. Stutzle, "Ant colony optimization," IEEE computational intelligence magazine, vol. 1, p. 28–39, 2006.
7. S. Mirjalili and A. Lewis, "The whale optimization algorithm," Advances in engineering software, vol. 95, p. 51–67, 2016.
8. E. Emary, H. M. Zawbaa and A. E. Hassanien, "Binary ant lion approaches for feature selection," Neurocomputing, vol. 213, p. 54–65, 2016.
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.
10. D. Karaboga and B. Akay, "A comparative study of artificial bee colony algorithm," Applied mathematics and computation, vol. 214, p. 108–132, 2009.
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.
13. E. Rashedi, H. Nezamabadi-Pour and S. Saryazdi, "GSA: a gravitational search algorithm," Information sciences, vol. 179, p. 2232–2248, 2009.
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.
15. A. Ahmadi-Javid, "Anarchic Society Optimization: A human-inspired method," in 2011 IEEE Congress of Evolutionary Computation (CEC), 2011.
16. M. O'Neill and C. Ryan, "Grammatical evolution," IEEE Transactions on Evolutionary Computation, vol. 5, p. 349–358, 2001.
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.
18. G. Manita and O. Korbaa, "Binary Political Optimizer for Feature Selection Using Gene Expression Data," Computational Intelligence and Neuroscience, vol. 2020, 2020.
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.
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.
21. M. M. Mafarja and S. Mirjalili, "Hybrid whale optimization algorithm with simulated annealing for feature selection," Neurocomputing, vol. 260, p. 302–312, 2017.
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.

23. M. Mafarja and S. Mirjalili, "Whale optimization approaches for wrapper feature selection," Applied Soft Computing, vol. 62, p. 441–453, 2018.
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.
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.
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.
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.
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.
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.
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.
32. M. A. Khanesar, M. Teshnehlab and M. A. Shoorehdeli, "A novel binary particle swarm optimization," in 2007 Mediterranean conference on control & automation, 2007.
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.
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.
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.
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.
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.

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.
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.
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.
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.
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.
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.
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.
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.
48. Hancer, Emrah. "An improved evolutionary wrapper-filter feature selection approach with a new initialisation scheme." *Machine Learning* (2021): 1-24.
49. Tsamardinos, Ioannis, et al. "A greedy feature selection algorithm for Big Data of high dimensionality." *Machine learning* 108.2 (2019): 149-202.
50. Shang, Ronghua, et al. "Unsupervised feature selection based on kernel fisher discriminant analysis and regression learning." *Machine Learning* 108.4 (2019): 659-686.
51. Singha, Sumanta, and Prakash P. Shenoy. "An adaptive heuristic for feature selection based on complementarity." *Machine Learning* 107.12 (2018): 2027-20