Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Arctic Puffin Algorithm with Local Escaping Operator for Feature Selection in High-Dimensional Datasets

Mona Selim

mona.selim@fci.suezuni.edu.eg

Suez University

Ahmed Fahim

Prince Sattam Bin Abdulaziz University

Dina A. Elmanakhly

Suez Canal University

Article

Keywords: Arctic Puffin Optimization algorithm, Feature selection, Metaheuristic Algorithm, High-dimensional data

Posted Date: March 12th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-8998105/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Arctic Puffin Algorithm with Local Escaping Operator for Feature Selection in High-Dimensional Datasets

Mona Selim[1,*], **Ahmed Fahim**[2,1], **and Dina A. Elmanakhly**[3]

[1]Department of Computer Science, Faculty of Computers and Information, Suez University, Suez, 43514, Egypt
[2]Computer Science Department, College of Computer Engineering and Science, Prince Sattam bin Abdulaziz University, Al-Kharj, Saudi Arabia
[3]Department of Mathematics, Faculty of Science, Suez Canal University, Ismailia, 44745, Egypt
[*]Corresponding author: Mona Selim (email: mona.selim@fci.suezuni.edu.eg)

ABSTRACT

Large datasets might make data mining more difficult due to their high dimensionality. To minimize the dimensionality of large datasets, feature selection is an essential step that maximizes classification accuracy by choosing the most informative features. In this study, a binary adaptation of the enhanced Arctic Puffin Optimization (EAPO) algorithm, which builds on the Arctic Puffin Optimization (APO) algorithm, is introduced within a wrapper-based framework to identify the optimal subset of features for classification tasks. The core component of EAPO lies in the design of the transfer function, which facilitates the transformation of the continuous search space into a binary domain. To evaluate its effectiveness, three transfer functions, S-shaped, V-shaped, and U-shaped, are investigated. Additionally, the local escaping operator is introduced as a technique to upgrade the performance of the APO algorithm via effectively balancing exploration and exploitation. The proposed EAPO method is assessed on 12 benchmark datasets using the k-Nearest Neighbor classifier as the evaluation metric. To control overfitting, k-fold cross-validation employs a partitioning strategy in which the dataset is divided into k folds, and each fold is used as the test set once while the remaining folds form the training set. Comparisons are made with state-of-the-art methods, including the butterfly optimization algorithm, sine cosine algorithm, Kepler optimization algorithm, grey wolf optimizer, cuckoo optimization method, and salp swarm algorithm. The results obtained from the experiments show that the EAPO technique is better in terms of consistency, feature reduction, and classification accuracy. Statistical analysis further confirms the effectiveness and robustness of the proposed method.

Keywords: Arctic Puffin Optimization algorithm, Feature selection, Metaheuristic Algorithm, High-dimensional data

1 Introduction

High-dimensional data are becoming increasingly prevalent as data collection methods are continuously updated and developed. However, a significant volume of redundant data introduces noise into the models used for pattern identification. This redundancy contributes to the model's complexity while also increasing its computational cost. Furthermore, it may occasionally direct the model's learning process in a way that produces a weak model, leading to inaccurate results. This could result in issues such as the curse of dimensionality and overfitting[1]. One well-established way of handling the classification issue of high-dimensional small-sample data is by reducing the dimensionality of the data[2]. Feature selection techniques are frequently employed to tackle these issues. The target of feature selection is choosing a few features as possible while adequately representing the entire feature space[3]. It can reduce the model's training time while preserving or improving classification accuracy.

Feature selection is regarded as an essential data preprocessing step in machine learning and pattern recognition as research advances and application fields expand[4]. However, in the current large-scale, high-dimensional data environment, traditional feature selection methods often fail to meet application requirements. Methods for attribute selection can be split into filter and wrapper approaches based on their relationship with the learning algorithm[5,6]. In the wrapper approach, one or more learning algorithms determine feature subsets. Although this method is computationally expensive, it outperforms the filter approach. Wrapper-based approaches are usually formulated as an optimization problem[7]. The filter-based approaches, on the other hand, select relevant features based on mutual information, information gain, and other criteria without relying on a learning algorithm[8,9]. While computationally efficient, this method generally performs worse than wrapper-based techniques. Since the aim is to utilize as few features as possible while obtaining maximum accuracy, identifying the most relevant subset is challenging. Selecting the optimal attribute set is time-consuming because attribute selection is an NP-hard problem[10]. Finding the optimal feature set requires evaluating $2^N - 1$ subsets for N features[11]. To reduce computation time, a high-performing metaheuristic is essential for addressing this problem.

Metaheuristic search methods are based on a compromise between diversification (exploration), which assesses the solution of candidates outside the neighborhood, and intensification (exploitation), which conducts a comprehensive neighborhood search to identify more feasible solutions. One's approach to discovering solutions is determined by these two factors. Because they perform better than exact methods, metaheuristic methods have recently been utilized to handle the attribute selection optimization problem[12],[13], and[14]. Metaheuristic algorithms may be an optimal choice as they can produce satisfactory solutions within an appropriate time framework. They can be a viable replacement to reduce the downside of extensive search that may be time-consuming and require high computational cost.

The Arctic Puffins (AP) predation and survival strategies served as an inspiration for the recently developed Arctic Puffin Optimization (APO) algorithm[15]. Three test sets—CEC2017, CEC2019, and CEC2022—are used to evaluate it, and the findings are quite encouraging. The APO authors state that the algorithm surpasses nine popular metaheuristic algorithms: the African vulture optimization Algorithm[16], the slime mould algorithm[17], the artificial gorilla troop optimizer[18], the reptile search algorithm[19], the arithmetic optimization algorithm[20], the covariance matrix adaptation evolution strategy[21], the runge-kutta optimization algorithm[22], the beluga whale optimization algorithm[23], and the Harris hawk optimization algorithm[24].

Despite extensive research, most metaheuristic algorithms continue to face several persistent challenges, such as a shortage of diversity, entrapment in local optima, and an imbalance between exploitation and exploration capabilities. Consequently, there remains a need for more advanced optimization strategies to achieve improved outcomes. Furthermore, as no single algorithm is able to resolve every optimization problem, the no-free-lunch theorem[25] highlights the importance of developing new or enhanced methods to generate high-quality solutions for specific problems. This motivated us to leverage the APO algorithm in feature selection problems because of its promising performance.

In this study, we propose an enhanced binary version of APO, termed Enhanced Arctic Puffin Optimization (EAPO) algorithm, to address the attribute selection challenge in high-dimensional datasets. EAPO integrates local escaping operator with APO to refine population variety and increase the algorithm's exploration efficiency, ensuring a better balance between exploitation and exploration. To gauge the performance of the picked attribute subsets, we employ k-Nearest Neighbors (kNN) as the classifier. The effectiveness of the proposed approach is illustrated using evaluations on 12 high-dimensional datasets, with feature counts varying from 2000 to 10367. Comparisons were conducted against seven traditional attribute selection methods. The findings reveal that EAPO outperforms the compared methods by finding the top-level average classification accuracy with the smallest subsets of picked attributes.

The key achievements of this work are outlined as follows:

- An enhanced binary version of the original APO algorithm, called EAPO, is proposed for feature selection. Within this framework, three transfer functions, S-shaped, V-shaped, and U-shaped, are evaluated to determine the most effective approach for mapping the continuous search space into a binary domain.
- The paper presents the local escaping operator as a tool to improve APO's performance by balancing exploitation and exploration.
- To show the efficacy of the proposed feature selection method, 12 high-dimensional datasets of various dimensionalities and sizes were subjected to extensive testing.
- To assess the relative superiority of the EAPO, many popular feature selection techniques are contrasted with it.
- To prove the superiority and efficacy of the proposed method, statistical analysis is conducted on the chosen high-dimensional datasets.

The subsequent sections of this manuscript are arranged as follows: Section 2 provides a review of relevant studies on the feature selection problem. Section 3 presents an overview of the APO algorithm. The proposed method is detailed in Section 4. In Section 5, we discuss the experimental setup and analyze the obtained results. The study is finally concluded in Section 6, which also delineates potential future research directions.

2 Related Work

In essence, metaheuristic methods may be categorized into four groups based on their inspiration: (a) evolutionary algorithms[26], (b) physics-based algorithms[27], (c) human-based algorithms[28], and (d) swarm intelligence[29].

The evolutionary behavior of organisms serves as a motivation for evolution-based algorithms. Notable examples include the Genetic Algorithm (GA)[30] and Differential Evolution (DE)[31]. In[32], a feature selection method is introduced that utilizes a community detection approach to form clusters of similar attributes, thereby reducing the selection of irrelevant features. Additionally, this technique uses a repair mechanism based on community detection, which considers both local and global

structures of the graph model to calculate similarity values. In[33], the authors propose an innovative hybrid genetic algorithm for feature selection, integrating local search to refine the search process. These operations are designed to enhance fine-tuning capabilities. The method offers two key advantages: significant improvement in final performance and improved subset size control. Additionally, the method demonstrates better convergence compared to traditional genetic algorithms.

Physics-based algorithms draw inspiration from various physical laws and principles, including those from physics, dynamic systems, music, chemistry, and metallurgical processes. For instance, in[34], a golden-ratio-based equilibrium optimization technique was suggested for attribute selection in speech emotion recognition, utilizing a merged version of two methods. The authors in[27] addressed the reactive power problem by combining tabu search and simulated annealing (SA) and applied this method to the symmetrical traveling salesman problem. The study in[35] introduced a hybrid feature selection method merging SA and GA, which demonstrated strong performance on the UCI dataset regarding the number of chosen features, outperforming state-of-the-art methods.

Human-based algorithms draw inspiration from human activities and behaviors that impact performance. Researchers develop these algorithms by observing such behaviors. One of the most notable methods in this area is Teaching–Learning-Based Optimizatio (TLBO)[36]. In[28], the authors employed Tsallis entropy-based feature selection to identify important features, utilizing a Support Vector Machine (SVM) for classification. They presented an improved version of teaching–learning-based optimization algorithm incorporating an adaptive weight strategy and the Kent chaotic map to optimize the both SVM and Tsallis entropy parameters. Another study[37] employed a binary version of the basic heap-based optimizer for solving the feature selection problem, using the kNN classifier to confirm the significance of the selected features.

Swarm intelligence approaches emulate the collective behavior of animals. Well-known techniques in this area are Particle Swarm Optimization (PSO)[38] and Grey Wolf Optimization (GWO)[39]. For feature selection,[40] introduced an algorithm inspired by the behavior of cuckoo birds searching for prey. The Ant Colony Optimization (ACO) algorithm, detailed in[41], utilizes ant search strategies to find optimal feature subsets and has demonstrated superior performance on breast cancer datasets compared to other methods. The authors in[42] suggested a strategy combining the Harris hawk optimizer with SVM for detecting Android malware. In[43], the authors developed two models hybridizing ACO and GA, comparing them against state-of-the-art methods on various UCI datasets. Another study[44] introduced a hybrid wrapper-based method combining GA and cuckoo search algorithms for feature selection. The authors in[45] introduced a local exploration paradigm guiding PSO to select relevant features based on their correlation with data. Additionally, in robotic manipulator applications, a modified PSO combined with DE was used as a case study for both sequential and parallel manipulators, as well as 10-DOF hybrid redundant serial-parallel robots[46].

In[47], the authors introduced an intelligent feature selection method utilizing GWO alongside the kNN classifier. This method, inspired by the natural hunting behavior of wolves, aims to identify the most relevant subset of features to improve the classification performance of emotion recognition systems. In[29], an Improved Binary Golden Jackal Optimization (IBGJO) method was proposed as a search strategy for wrapper-based attribute selection. The IBGJO algorithm has three key features: (1) a population initialization step using a chaotic tent map to enhance exploitation abilities and guarantee the diversity of the population, (2) cosine similarity is used in an adaptive position update strategy to prevent premature convergence, and (3) a binary method created especially for problems involving the selection of binary features. In[48], the authors presented an approach for dimensionality reduction and informative gene selection using fractional calculus concepts. They introduced an adjusted version of the binary bat algorithm called the Fractional-Order Binary Bat Algorithm (FBBA), which controls the convergence process by utilizing greater historical recollection of bat behavior. The gene selection technique employs a two-stage hybrid filter/wrapper method, incorporating a new correlation-based feature clustering algorithm in the filter stage and the FBBA in the wrapper stage. The authors in[49] improved the zebra optimization algorithm by integrating a logarithmic spiral strategy, chaotic mapping, and an enhanced search quality mechanism for the feature selection problem.

To the best of our knowledge, no prior research has explored the binary adaptation of APO for the feature selection problem. We selected APO with the goal of developing and enhancing a binary version specifically designed for feature selection, motivated by its proven effectiveness across a range of optimization tasks, particularly in prediction and classification domains[50–52]. This identified gap in the literature has driven us to propose a novel binary APO variant, which we will rigorously evaluate as a robust binary optimization method for feature selection.

3 Proposed Method

3.1 Arctic Puffin Optimization: An overview

The APO is a metaheuristic optimization method that draws inspiration from AP predation and survival strategies. The APO is divided into three phases: initialization of the population, exploration (airborne flight) and exploitation (underwater foraging). Furthermore, it includes a behavioral transformation factor β to switch among these two phases. The following describes the mathematical models for each phase.

3.1.1 Initial Population

AP are constantly working together and migrating in groups. In the optimization process, each AP refers to a potential solution. The population initialization generation process is described by the following equation:

$$\vec{S}_i^t = \text{rand} \times (ub - lb) + lb, \quad i = 1, 2, 3, \ldots, N \tag{1}$$

where $\vec{S}_i^t$ denotes the location of the ith AP, N is the total count of the population, and *rand* produces a random value in (0, 1). Also, lb and ub stand for the lower and upper bounds, respectively.

3.1.2 Exploration Phase

AP use special techniques of flight and foraging. They smoothly move between the air and the ocean. When flying, puffins use primary key strategies to handle various circumstances. Searching from the air is the first strategy, and scuba diving for prey is the second. The various behavioral strategies of these two stages demonstrate how AP can adapt to diverse circumstances, which aids them surviving and procreating. This subsection will go into great depth about these two strategies and explain their importance in AP's existence.

Aerial search

AP usually fly in groups or formations as a cooperative behavior that maximizes the efficiency of flight and offers chances for hunting cooperatively. They fly at comparatively low altitudes to capture potential undersea aliment sources. While keeping vigilant for potential predators nearby, they choose to scout possible prey throughout this stage. When conditions are good, particularly when there is few predators and abundance, they can quickly migrate to the water surface to properly collect the food. The equations for position updates related to this approach are as follows.

$$\vec{Z}_i^{t+1} = \vec{S}_i^t + \left(\vec{S}_i^t - \vec{S}_r^t\right) * L(D) + R \tag{2}$$

$$R = \text{round}(0.5 * (0.05 + \text{rand}) * \chi) \tag{3}$$

$$\chi \sim \text{Normal}(0, 1) \tag{4}$$

where r represents a random number, with $1 \leq r \leq N-1$ and $r \neq i$. $\vec{S}_i^t$ refers to the current i^{th} candidate solution in the population, whereas $\vec{S}_r^t$ is a randomly picked candidate solution from the current population with $\vec{S}_i^t \neq \vec{S}_r^t$. Furthermore, χ is a random number from a typical normal distribution and $L(D)$ refers to a random number created via Lévy flight, where D represents the dimensionality.

AP uses the Lévy flight coefficient, where it functions similarly to their powerful wings, for changing its location throughout the aerial search technique. Known for its distinctive long-distance leaps, Levy flight allows puffins to effectively search for food sources[53,54]. They can quickly cover large maritime areas and continuously stay near possible aliment sources and efficiently hunt for food supplies because of this flight search approach. At the same time, while facing avian predators like seagulls, puffins utilize a circular flight method to fend off enemies, making it challenging for enemies to succeed. Finding appropriate fishing grounds is a key tactic in the puffin aerial search technique. The ability of the algorithm to search globally is improved by the flying strategy used during this phase, which increases the probability of reaching the globally optimal solution.

Predatory Swoop Behavior

Swooping is a key tactic that AP utilize to quickly reroute their flight in order to grab food more quickly. In order to survive against rivals, they must guarantee faster and effective prey catching, which makes this swooping strategy crucial. A velocity coefficient S was added to simulate this swooping behavior by modifying the puffin's displacement during the swoop. Hence, the position update equation is computed using the following equation.

$$\vec{X}_i^{t+1} = \vec{Z}_i^{t+1} \cdot C \tag{5}$$

$$C = \tan(\pi \cdot (\text{rand} - 0.5)) \tag{6}$$

The AP modify its movement in the initial phase by adding a velocity coefficient C, and π is a mathematical constant approximately equal to 3.14. C is crucial for regulating the AP's flying path and pace. C is a mathematical velocity coefficient that, by modifying the value and direction of its flight speed, enables the puffin to adjust to various feeding requirements. An increase in C enhances the algorithm's adaptability, allowing it to cope with competition, uncertainty, and the challenges of a more complex aerial environment. This results from the algorithm's ability to accurately simulate the puffin's aerial behavior. The algorithm's unpredictability and diversity are also increased by adding the distributional features of the parameter C, which also improves the AP's capacity for exploration. As a result, performing the speed predation stage improves the algorithm's effectiveness in exploring the solution space, enhancing its adaptability and search performance in many scenarios.

Two main behavioral techniques are generally used in the APO's flight stage to enhance the algorithm's exploration stage. The algorithm employs the Lévy flight coefficient to enhance global search in the aerial flying strategy. The algorithm broadens the scope of the investigation by adding more variation and randomization through the Lévy flight coefficient. This approach improves the APO's capacity to search widely for viable solutions by allowing it to thoroughly examine the solution space. Furthermore, the algorithm makes use of the swooping predation method to find solutions via quicker searches after identifying the appropriate hunting grounds. The swooping predation strategy enhances the algorithm's ability to explore the solution space efficiently. By focusing rapidly and accurately on promising candidates, the APO improves both the speed and precision of the search, facilitating the identification of high-quality solutions in less time.

The algorithm integrates the candidate positions generated in both phases to create a new solution, aiming to achieve optimal performance across various scenarios. The best N candidates are selected after sorting these solutions by fitness to generate the new population. This can be described using the following equations.

$$\vec{A}_i^{t+1} = \vec{Z}_i^{t+1} \cup \vec{X}_i^{t+1} \tag{7}$$

$$\text{new} = \text{sort}(\vec{A}_i^{t+1}) \tag{8}$$

$$\vec{S}_i^{t+1} = \text{new}(1:N) \tag{9}$$

where *sort* arranges the new population according to their fitness rankings, ascendingly, for choosing the next population $\vec{S}_i^{t+1}$.

3.1.3 Exploitation Phase

Aerial flight and underwater foraging are two essential components of the AP's survival strategy. Three primary techniques (gathering foraging, intensifying search, and avoiding predators) make up underwater foraging. Each technique is used in a particular situation to increase the effectiveness of predation. The following is a detailed description of these techniques.

Gathering foraging

AP frequently uses a collective approach when foraging, congregating to surround schools of fish close to the water's surface. Through this cooperative predation approach, puffins effectively surround and catch groups of fish, increasing the efficiency and likelihood of being successful in their predation. In order to locate diving hotspots or food sources, puffins on the water surface also observe how other members behave. The position update is done using this equation.

$$\vec{W}_i^{t+1} = \begin{cases} \vec{S}_{r1}^t + G \cdot L(D) \cdot (\vec{S}_{r2}^t - \vec{S}_{r3}^t) & \text{if rand} \geq 0.5 \\ \vec{S}_{r1}^t + G \cdot (\vec{S}_{r2}^t - \vec{S}_{r3}^t) & \text{if rand} < 0.5 \end{cases} \tag{10}$$

where candidate solutions $\vec{S}_{r1}^t$,$\vec{S}_{r2}^t$,$\vec{S}_{r3}^t$ were chosen at random from the current population, and $\vec{S}_{r2}^t \neq \vec{S}_{r3}^t$. Furthermore, $r1, r2, r3$ are unspecified integers from 1 to $N-1$ (excluding i), $r1 \neq r2 \neq r3$. Wheras G is the cooperative factor to adapt AP'predation behavior, and is used here as $G = 0.5$.

To illustrate foraging behavior, the gathering foraging approach is described by two position equations. According to these equations, AP use the cooperative factor G to forage cooperatively with other puffins when $rand < 0.5$. They roam randomly throughout their surroundings in search of foraging or other resources. If $rand \geq 0.5$, the other equation depicts a more intricate approach to food-seeking. When a puffin spots a school of fish during the gathering foraging process, it may follow other participants and quickly swim toward a better favorable predation team for improved cooperative foraging. Once again, the puffin's ability to acquire prey quickly and effectively is enhanced by the Levy flight factor. In optimization, this dual strategy technique aids in striking an equilibrium between exploitation and exploration, which causes the solution space to be searched more thoroughly.

The parameter G in the APO algorithm serves as a synergistic element, inspired by the collaborative foraging behavior of AP. These puffins frequently gather around fish near the water's surface, using an aggregation foraging strategy that significantly boosts their efficiency and success. By emulating this cooperative behavior, the parameter G aligns the algorithm more closely with natural animal behaviors. Setting the G value to 0.5, as determined through sensitivity analysis, enhances performance and solution diversity by balancing exploration and development processes.

Intensifying search

While predation persists, AP may eventually sense a decrease or depletion of food availability in their current foraging area. To continue obtaining the nutrients they require, they must change their underwater places to look for other underwater food sources or additional fish. Next is the location update formula for this stage:

$$\vec{Z}_i^{t+1} = \vec{W}_i^{t+1} \cdot (1+g) \tag{11}$$

$$g = 0.1 \cdot (rand - 1) \cdot \frac{IT - t}{IT} \tag{12}$$

where t denotes iterations currently used, and IT is the total count of iterations. Also, *rand* adds a degree of randomness to g by being a random number. This stage is described by Eq. 11, which represents modifications made to the initial location, the parameter $(1+g)$, where the adaptive factor g is utilized to change the AP's location in the water, g is crucial in the second step of the improved search method. The adaptability of puffins in adjusting to their surroundings while foraging served as inspiration for the design of this adaptive element. In particular, the parameter g is progressively changed as the number of iterations rises, allowing the puffin to make decisions about shifting its location in response to the progress and unpredictability of its hunt for more plentiful food sources.

By using this technique, AP can meet their survival demands during predation by flexibly adapting to the distribution of food in their surroundings. The algorithm's ability to adapt and grow in the face of changing challenges and innovative solutions is facilitated by its dynamic exploration of the search space to uncover new solutions.

Avoiding predators

When AP identifies predators nearby, they use this approach to explain their behavior. To warn other puffins of impending danger, they make a particular sound or call. Other puffins become alert and move away from the dangerous area as a result of this call, which acts as a danger signal. Concurrently, when AP sense a predator approaching, they quickly change their location, swimming quickly along a wider trajectory in the direction of safety to avert the threat. The location-altering equation for this technique is as follows:

$$\vec{X}_i^{t+1} = \begin{cases} \vec{S}_i + \beta \times (\vec{S}_{r1} - \vec{S}_{r2}) & \text{if rand} < 0.5 \\ \vec{S}_i + G \times L(D) \times (\vec{S}_{r1} - \vec{S}_{r2}) & \text{if rand} \geq 0.5 \end{cases} \tag{13}$$

where β is a random number uniformly distributed between 0 and 1. The approach incorporates an excellent balancing procedure, particularly in dangerous circumstances. This includes two behavioral modes: one for quickly dodging danger and the other for evading it. The algorithm escapes from local optima in various circumstances by employing this dual-behavior approach. When $rand \geq 0.5$, meaning a nearby predator, AP rapidly dodges, significantly altering their position. This is a way of using existing knowledge to improve jumping capacity, hence guiding the algorithm toward superior solutions in feasible local optima. Conversely, when $rand < 0.5$, AP proactively evades predators with more cautious behavior, randomly altering their positions. This exploratory method allows the algorithm to search the surrounding environment carefully, aiming to get the best solutions.

This dual-behavior mechanism enhances the algorithm's capacity to maintain balance across diverse conditions. By carefully exploring new regions and rapidly incorporating existing knowledge, it strengthens the algorithm's global search ability and reduces the likelihood of stagnation in local optima. The algorithm simulates this balancing mechanism to guarantee its adaptability and robustness.

In conclusion, puffins use diverse techniques when underwater foraging, such as intense searching, avoiding predators, and gathering forage. These strategies could result in varoius foraging outcomes depending on the different circumstances. To obtain the best results in different circumstances, the method selects to combine the candidate locations of the three distinct location equations to a new solution. The highest N solutions are chosen after all the solutions are arranged according to their fitness. This can be described using the following equations.

$$\vec{A}_i^{t+1} = \vec{W}_i^{t+1} \cup \vec{Z}_i^{t+1} \cup \vec{X}_i^{t+1} \tag{14}$$

$$\text{new} = \text{sort}(\vec{A}_i^{t+1}) \tag{15}$$

$$\vec{S}_i^{t+1} = \text{new}(1:N) \tag{16}$$

This all-encompassing approach enables the model to take into account several foraging scenarios at once, finally choosing the best position to be the optimal solution. This improves the puffins' ability to forage and survive in a variety of circumstances.

3.1.4 Conversion factor for behavior β

In the APO method, AP initially carry out frequent aerial flights for achieving global search and later dive for food for achieving the local exploitation. This behavior is motivated by their life routines, where they first search for feeding waters and then dive for food. According to this behavioral design, APO uses the coefficient of behavioral switching β to smoothly switch from universal search to regional exploitation, which can be defined using the following equation.

$$\beta = 2 * \log(1/rand) * (1 - t/IT) \tag{17}$$

where *rand* represents an unspecified number between 0 and 1. Furthermore, β in the APO algorithm is a dynamically changing parameter based on the ratio of current to maximum iteration number. It also includes randomness to emulate the unpredictable nature of AP behavior. This design permits dynamic adjustments to β throughout iterations, meeting search requirements at various stages.

The APO algorithm uses parameter V (set to 0.5), which is compared with β to choose a search technique. When β exceeds V, the algorithm conducts a global search (air flight phase) to explore a large area of the solution space and avoid early local optima. If β is smaller than or equal to V, the method executes a local search (diving and foraging phase) to refine the accuracy of the high-quality solutions found immediately. This approach balances exploration and exploitation to improve convergence speed and algorithm efficiency.

Therefore, the behavioral switching coefficient β is crucial for balancing exploration and exploitation within the algorithm. It mirrors the transitions among two life phases of the AP: underwater foraging and aerial flight. By comparing β with V, the algorithm can dynamically adjust its search approach based on various iteration steps. This design enhances the algorithm's efficiency and robustness by balancing local exploitation with global search.

3.2 Enhanced Arctic Puffin Algorithm

We propose to enhance the APO algorithm by incorporating local escape operator. This operator was introduced as a regional search approach in[56] to enhance the optimization algorithm in exploring new areas required for complex problems in the real world. It improves solution quality by updating positions based on specific criteria, preventing the method from getting stuck in local optima and enhancing convergence. It produces a substitute solution (S_{LEO}) by using various solutions like the best position S_{best}, two randomly generated solutions S^m_{r1} and S^m_{r2}, and a newly generated solution S^m_k. The solution S_{LEO} is determined as:

If $r2 < 0.4$ Then

$$S^m_{\mathrm{LEO}} = S^m_n + g_1\left(f_1 S_{\mathrm{best}} - f_2 S^m_k\right) + g_2\rho_1(f_3(S2^m_{\mathrm{n}} - S1^m_{\mathrm{n}}) + f_2(S^m_{\mathrm{r1}} - S^m_{\mathrm{r2}})/2 \tag{18}$$

$$S^{m+1}_n = S_{LEO} \tag{19}$$

Else

$$S^m_{\mathrm{LEO}} = S_{\mathrm{best}} + g_1\left(f_1 S_{\mathrm{best}} - f_2 S^m_k\right) + g_2\rho_1\left(f_3(S2^m_{\mathrm{n}} - S1^m_{\mathrm{n}}) + f_2(S^m_{\mathrm{r1}} - S^m_{\mathrm{r2}})/2\right). \tag{20}$$

$$S^{m+1}_n = S_{LEO} \tag{21}$$

End

where S^m_n is the existing position, r_2 is a random number in the range $[0,1]$, g_1 and g_2 are random values in the range $[-1,1]$ that are uniformly distributed. Also, $S1^m_n$ and $S2^m_n$ are a pair of random solutions taken from the current population and are defined as:

$$S1^m_n,\ S2^m_n = lb + rand(\mathrm{D}) \times (ub - lb) \tag{22}$$

where D is the dimension of any solution. Moreover, n and m are the coordinates of the solution ($n = 1,2,3,\ldots,N$) and ($m = 1,2,3,\ldots,\mathrm{D}$). In addition, f_1, f_2, and f_3 are three random variables generated as:

$$f_1 = U_1 \times 2 \times rand + (1 - U_1) \tag{23}$$

$$f_2 = U_1 \times rand + (1 - U_1) \tag{24}$$

$$f_3 = U_1 \times rand + (1 - U_1) \tag{25}$$

where U_1 is a binary parameter ($U_1 = 1$ if $\Psi_1 < 0.5$, and 0 otherwise), Ψ_1 is a value in the range of $[0,1]$.

Furthermore, ρ_1 aims to balance between exploitation and exploration in searching operations, and it may be computed as:

$$\rho_1 = 2 \times rand \times \xi - \xi \tag{26}$$

$$\xi = \left|\eta \times \sin\left(\frac{3\pi}{2} + \sin\left(\eta \times \frac{3\pi}{2}\right)\right)\right| \tag{27}$$

$$\eta = \eta_{\min} + (\eta_{\max} - \eta_{\min}) \times \left(1 - \left(\frac{t}{IT}\right)^3\right)^2 \tag{28}$$

where $\eta_{\max}$ and $\eta_{\min}$ are set to 1.2 and 0.2, respectively. To achieve a balance between the processes of exploitation and exploration, ρ_1 varies according to the sine function ξ.
The solution S_k^m in Eqs. (18) and (20), can be deriven as follows:

$$S_k^m = \begin{cases} S_{\text{rand}} & \text{if } \Psi_2 < 0.5 \\ S_p^m & \text{otherwise} \end{cases} \tag{29}$$

where S_{rand} is a new solution, computed as in Eq. (30), S_p^m is a random solution taken from the population ($p \in [1,2,\ldots,N]$), Ψ_2 is a random value in $[0,1]$.

$$S_{\text{rand}} = lb + \text{rand}(0,1) \times (ub - lb) \tag{30}$$

Eq. (29) may be rewritten as in Eq (31)

$$S_k^m = U_2 \times S_p^m + (1 - U_2) \times S_{\text{rand}} \tag{31}$$

where U_2 is a binary parameter. If parameter Ψ_1 is less than 0.5, the value of U_2 is 1, otherwise, it is 0.

The proposed EAPO algorithm extends the standard APO by incorporating the local escape operator procedure, thereby improving its exploration and exploitation capabilities. The detailed steps of the EAPO algorithm are presented in Algorithm 1.

3.3 Transformation Function

Following the approach proposed in[55], a common method for adapting a continuous optimization algorithm to a binary problem is to employ a transfer function (TF) that maps real-valued positions to binary decisions without modifying the algorithm's fundamental structure. The TF outputs a probability that determines whether a feature should be selected (1) or not (0). In this study, we evaluated three different TF, S-shaped, V-shaped, and U-shaped, to identify the most effective one for APO, since no single TF has yet been recognized as superior. These functions are presented below:

$$S(x) = \frac{1}{1 + e^{-x/2}} \tag{32}$$

$$V(x) = \left| \frac{2}{\pi} \arctan\left(\frac{\pi}{2} x\right) \right| \tag{33}$$

$$U(x) = \tau \left| x^{\frac{3}{2}} \right| \tag{34}$$

where τ in Eq. (34) is set to 1 in this study. To demonstrate the conversion of a continuous search space into a binary domain, a U-shaped function was selected as a representative example of the U-shaped family:

$$T(s_{id}^t) = \tau |(s_{id}^t)^{3/2}| \tag{35}$$

Let s_{id}^t represent the position of the ith individual in the dth dimension at iteration t. A continuous value of s_{id}^t is passed through a TF to produce a probability, which is then compared to a uniformly generated random number $r \in [0,1]$ to make the binary selection. This function generates a probability value in the range [0, 1], indicating the likelihood of selecting a feature. The corresponding binary decision rule is defined in Eq. (36):

$$s_{id}^{t+1} = \begin{cases} 1, & \text{if } r \geq T(s_{id}^t) \\ 0, & \text{otherwise} \end{cases} \tag{36}$$

This process is applied to each dimension of every individual, enabling the transformation of continuous positions into binary feature selection masks. A lower value of $T(s_{id}^t)$ implies a lower probability of selecting the corresponding feature in the next iteration, promoting sparsity in the selected feature subset.

Algorithm 1: EAPO Algorithm

Input: N, IT, D, G, and V
Output: the best $\vec{S}_i^t$ and its fitness value
Initialize N puffins, $\vec{S}_i^t$ ($i = 1, 2, \ldots, N$), using Eq. (1)
Evaluate each $\vec{S}_i^t$ and find the one with the best fitness in $\vec{S}_i^t$
$t \leftarrow 1$
while $t < IT$ **do**
    Calculate the behavior conversion factor β according to Eq. (17)
    if $\beta > V$ {Air flight period (Exploration phase)}
        for $i = 1$ **to** N
            $\vec{Z}_i^{t+1}$ using Eq. (2) {Air search}
            Update $\vec{X}_i^{t+1}$ using Eq. (5) {Dive predation}
            Select N excellent populations as the new population $\vec{S}_i^{t+1}$ using Eq. (7)-(9)
            Evaluate the puffins, $\vec{S}_i^{t+1}$, and replace $\vec{S}_i^t$ with $\vec{S}_i^{t+1}$ if it is better.
            $t \leftarrow t + 1$
        end for
    else {Diving and foraging period (Exploitation phase)}
        for $i = 1$ **to** N
            Update $\vec{W}_i^{t+1}$ using Eq. (10) {Gather for food}
            Update $\vec{Z}_i^{t+1}$ using Eq. (11) {Strengthen search}
            Update $\vec{X}_i^{t+1}$ using Eq. (13) {Avoid predators}
            Select N excellent populations as the new population $\vec{S}_i^{t+1}$ using Eq. (14)-(16)
            Evaluate the puffins, $\vec{S}_i^{t+1}$, and replace $\vec{S}_i^t$ with $\vec{S}_i^{t+1}$ if it is better.
            Update $\vec{S}_i^{t+1}$ based on local escaping operator using Eqs. 18, 19, or Eqs. 20, 21,
            Evaluate the puffins, $\vec{S}_i^{t+1}$, and replace $\vec{S}_i^t$ with $\vec{S}_i^{t+1}$ if it is better.
        end for
        $t \leftarrow t + 1$
    end if
end while
return $\vec{S}_i^t$

4 Experiments

All experiments were conducted using Matlab software on Windows 10 with an Intel Core i7-3700 (3.4 GHz) and 16 GB of RAM.

4.1 Evaluation Measures

The proposed feature selection technique is evaluated using several performance metrics, including fitness function values, accuracy, macro precision, macro sensitivity, and the number of selected features. The definitions and descriptions of each of these assessment measures are provided below:

- **Accuracy**: This metric evaluates the model's ability to correctly classify instances, which indirectly reflects the effectiveness of the selected features.

$$\text{Accuracy} = 1 - \text{Error_Rate} \tag{37}$$

- **Fitness Function Value**: This metric assesses the overall quality of a solution by balancing classification performance with the number of selected features. The fitness function value is computed as defined in Eq. (40).
- **Number of Selected Features**: This metric tracks the number of features selected by the proposed method in each solution, providing insight into the compactness and efficiency of the feature subset produced.
- **Macro Precision**: Precision measures the proportion of correctly predicted positive instances to the total predicted positive instances. Macro precision is the average of the precision scores computed for each class.

$$\text{Macro Precision} = \frac{1}{Cl}\sum_{i=1}^{Cl}\frac{TP_i}{TP_i + FP_i} \tag{38}$$

  where TP_i and FP_i are the number of true positives and false positives, for class i. Also, Cl is the number of classes.

- **Macro Sensitivity**: Also known as macro recall. Sensitivity measures the proportion of correctly predicted positive instances to all actual positive instances. Macro sensitivity is the average of the sensitivity scores computed for each class.

$$\text{Macro Sensitivity} = \frac{1}{Cl}\sum_{i=1}^{Cl}\frac{TP_i}{TP_i + FN_i} \tag{39}$$

  where FN_i is the number of false negatives for class i.

- **Average computational time**: indicates the average run time, in minutes, of an individual optimization algorithm across multiple runs.

To evaluate the feature selection datasets using the proposed binary method, the data is randomly divided into training and testing sets through 5-fold cross-validation to ensure robust results. The evaluation focuses on two primary objectives: reducing the number of selected features and enhancing classification accuracy. This multi-objective approach combines the two goals according to the fitness criterion, with a specified weight factor as outlined in Eq. 40.

$$\text{fitness} = a \cdot \text{Error_Rate} + b \cdot \frac{|R|}{|D|} \tag{40}$$

where Error_Rate represents the classification error rate of the dataset D obtained using a predefined classifier. $|D|$ indicates the total number of features in the dataset, while $|R|$ denotes the number of selected features. a and b are weighting factors that trade off between classification accuracy and feature subset size. The values of these parameters are given as $a \in [0,1]$ and $b = 1 - a$.

The fitness function in Eq. 40 plays a central role in any wrapper-based feature selection approach. Two critical aspects must be considered: (a) how candidate solutions are represented and (b) how they are evaluated. In this work, binary vectors are used to represent feature subsets, where the vector length corresponds to the total number of features in the dataset. Each solution is evaluated using both the EAPO and APO algorithms. The key objectives of a wrapper-based feature selection method are to minimize the number of selected features and maximize classification accuracy. In this study, a kNN classifier is used to define the fitness function and assess the quality of the selected subsets.

4.2 Datasets

Twelve benchmark datasets were taken from the two sources [1] [2], to evaluate the effectiveness of the proposed approach. These datasets were chosen for their variation in features number and instances, providing diverse challenges for evaluating the approach. Additionally, we incorporated a collection of high-dimensional datasets to assess the EAPO algorithm's performance in large search spaces. The datasets used are summarized in Table 1, which vary in the number of classes (ranging from 2 to 10), instances (ranging from 50 to 130), and features (ranging from 2000 to 10367).

Table 1. Overview of the benchmark datasets utilized.

#	Dataset	Attributes	Instances	Classes
1	Colon	2000	62	2
2	WarpAR10P	2400	130	10
3	Lymphoma	4026	96	9
4	GLIOMA	4434	50	4
5	Leukemia_1	5327	72	3
6	DLBCL	5469	77	2
7	Brain_Tumor_1	5920	90	5
8	Prostate_GE	5966	102	2
9	Prostate_Tumor_1	5966	102	2
10	ALLAML	7129	72	2
11	Leukemia_2	7129	72	4
12	Brain_Tumor_2	10367	50	4

4.3 Configuration of Parameters

The performance of EAPO is assessed against several recognized feature selection techniques, summarized as follows: Butterfly Optimization Algorithm (BWO)[57], Sine Cosine Algorithm (SCA)[58], Kepler Optimization Algorithm (KOA)[59], Cuckoo Optimization Algorithm (COA)[60], Salp Swarm Algorithm (SSA)[61], and GWO[39].

Every algorithm is run 20 times independently. For all subsequent experiments, the upper limit of iterations is set to 70, with a population size of 15 search agents. Under the guidance of the literature, a number of tests were carried out on various datasets to ascertain the ideal values for a and b. Consequently, a is set to 0.99 and b to 0.01.

The study evaluates the performance of various algorithms on feature selection problems compared to the proposed approach. The kNN classifier is used as the wrapper method because of its simplicity, and it is easy to implement. Unlike other classifiers, kNN requires tuning only one parameter k. After multiple trials on different datasets, the optimal value for k was determined to be 5. The dataset is split into five folds using the 5-fold cross-validation method, with training and testing occurring at a 4:1 ratio. This approach minimizes the variance in accuracy across folds. The kNN classifier is trained using the training data, whereas the test data remains withheld during the training phase.

4.4 Experimental results

The experimentation process consists of three distinct stages. In the initial stage, the impact of three distinct TFs on the original APO is examined to identify the most suitable one for integration into the proposed method. In the subsequent phase, the proposed EAPO algorithm is benchmarked against the original APO to assess its effectiveness and the extent of its improvements. In the third stage, EAPO is further assessed by benchmarking it against other competitive wrapper-based feature selection methods, providing a comprehensive evaluation of its performance.

4.4.1 First phase: Comparative analysis of APO variants

In this section, we analyze the performance of three TFs combined with APO (APO-S, APO-V, and APO-U) for the feature selection problem. The evaluation is carried out using best, average, and worst values for three criteria: fitness, classification accuracy, and the number of selected features.

The best, mean, and worst fitness values for all APO variations are presented in Table 2. The best values are highlighted in bold for emphasis. **APO-U** ranked first in six out of the 12 datasets (50%) across these measures, specifically for *colon*, *warpAR10P*, *Leukemia_1*, *Brain_Tumor_1*, *Brain_Tumor_2*, and *Leukemia_2*. **APO-S** achieved the top results in four datasets (*lymphoma*, *GLIOMA*, *DLBCL*, and *Prostate_GE*), representing approximately 33% of the datasets. Finally, **APO-V** obtained the best performance in only two datasets (*ALLAML_M* and *Prostate_Tumor_1*), accounting for about 17% of the datasets.

[1] https://ckzixf.github.io/dataset.html
[2] https://jundongl.github.io/scikit-feature/datasets.html

The best, mean, and worst classification accuracy results are summarized in Table 3. **APO-U** achieved first place in seven datasets: *colon*, *warpAR10P*, *Leukemia_1*, *Brain_Tumor_1*, *Brain_Tumor_2*, *Leukemia_2*, and *lymphoma*, representing 58% of the datasets. **APO-S** ranked first in three datasets: *DLBCL*, *GLIOMA*, and *Prostate_GE*, and shared the top performance with APO-U in *lymphoma*. Finally, **APO-V** achieved the best results in two datasets (*ALLAML_M* and *Prostate_Tumor_1*), and matched APO-S in *GLIOMA*.

The best, mean, and worst number of selected features for all APO variations are presented in Table 4. The results indicate that **APO-S** outperforms the other variations in this criterion, ranking first in all 12 datasets, followed by **APO-V**, and then **APO-U**. Moreover, Fig 1 illustrates the mean number of features chosen across the datasets.

The results of the preceding analysis indicate that the TF **APO-U** is the most effective variation in terms of fitness and classification accuracy, while **APO-S** excels in reducing the number of features. Although dimensionality reduction is an important goal, the primary objective remains to maximize classification accuracy while simultaneously reducing dimensionality. As **APO-U** consistently outperforms the other variants in terms of best, mean, and worst fitness values, as well as best, mean, and worst classification accuracy, it is identified as the best-performing variant of the APO algorithm.

Table 2. A comparative analysis of APO variations in terms of their best, average, and worst fitness values.

#	Dataset	Best fitness			Average fitness			Worst fitness		
		APO-U	APO-V	APO-S	APO-U	APO-V	APO-S	APO-U	APO-V	APO-S
1	colon	**0.1659**	0.1961	0.1808	**0.1829**	0.2136	0.1854	**0.1965**	0.2292	0.1961
2	warpAR10P	**0.4249**	0.4324	0.4994	**0.4299**	0.4442	0.5052	**0.4328**	0.4551	0.5072
3	lymphoma	0.0566	0.0882	**0.0559**	0.0575	0.0899	**0.0559**	0.0576	0.0986	**0.0560**
4	GLIOMA	0.1435	0.1244	**0.1231**	0.1438	0.1246	**0.1232**	0.1444	0.1247	**0.1232**
5	Leukemia_1	**0.0195**	0.0340	0.0467	**0.0326**	0.0369	0.0628	**0.0457**	0.0483	0.0873
6	DLBCL	0.0580	0.0704	**0.0430**	0.0667	0.0709	**0.0469**	0.0828	0.0710	**0.0563**
7	Brain_Tumor_1	**0.1049**	0.1268	0.1692	**0.1127**	0.1319	0.1693	**0.1160**	0.1380	0.1693
8	Prostate_GE	0.1026	0.1030	**0.0915**	0.1051	0.1129	**0.0980**	0.1126	0.1218	**0.1015**
9	ALLAML_M	0.1300	**0.1296**	0.1551	0.1408	**0.1403**	0.1644	0.1445	**0.1439**	0.1684
10	Brain_Tumor_2	**0.1642**	0.1840	0.2220	**0.1792**	0.1970	0.2221	**0.1843**	0.2042	0.2221
11	Prostate_Tumor_1	0.1330	**0.0925**	0.1401	0.1343	**0.0937**	0.1481	0.1433	**0.1026**	0.1500
12	Leukemia_2	**0.1154**	0.1177	0.1279	**0.1226**	0.1335	0.1416	**0.1295**	0.1447	0.1553

Table 3. A comparative analysis of APO variations in terms of their best, average, and worst classification accuracies.

#	Dataset	Best accuracy			Average accuracy			Worst accuracy		
		APO-U	APO-V	APO-S	APO-U	APO-V	APO-S	APO-U	APO-V	APO-S
1	colon	**0.8385**	0.8077	0.8218	**0.8214**	0.7901	0.8172	**0.8077**	0.7744	0.8064
2	warpAR10P	**0.5769**	0.5692	0.5000	**0.5719**	0.5573	0.4942	**0.5692**	0.5462	0.4923
3	lymphoma	**0.9479**	0.9168	**0.9479**	**0.9479**	0.9153	**0.9479**	**0.9479**	0.9063	**0.9479**
4	GLIOMA	0.8600	**0.8800**	**0.8800**	0.8600	**0.8800**	**0.8800**	0.8600	**0.8800**	**0.8800**
5	Leukemia_1	**0.9867**	0.9714	0.9571	**0.9733**	0.9687	0.9410	**0.9600**	0.9571	0.9162
6	DLBCL	0.9475	0.9342	**0.9608**	0.9388	0.9342	**0.9570**	0.9225	0.9342	**0.9475**
7	Brain_Tumor_1	**0.9000**	0.8778	0.8333	**0.8922**	0.8728	0.8333	**0.8889**	0.8667	0.8333
8	Prostate_GE	0.9024	0.9019	**0.9119**	0.9000	0.8919	**0.9054**	0.8924	0.8829	**0.9019**
9	ALLAML_M	0.8743	**0.8752**	0.8476	0.8637	**0.8644**	0.8383	0.8600	**0.8610**	0.8343
10	Brain_Tumor_2	**0.8400**	0.8200	0.7800	**0.8250**	0.8070	0.7800	**0.8200**	0.8000	0.7800
11	Prostate_Tumor_1	0.8719	**0.9124**	0.8629	0.8705	**0.9114**	0.8548	0.8614	**0.9024**	0.8529
12	Leukemia_2	**0.8895**	0.8876	0.8752	**0.8823**	0.8714	0.8614	**0.8752**	0.8600	0.8476

4.4.2 Second phase: Effectiveness of the Local Escaping Operator within the proposed method.

In a subsequent experiment, we focus on investigating the impact of incorporating the local escaping operator into APO. Various measures are employed to analyze the performance of APO and EAPO in the feature selection problem:

- Fitness measures, including the best, average, and worst values

Table 4. A comparative evaluation of APO variations based on the best, average, and worst counts of selected features.

#	Dataset	Best count			Average count			Worst count		
		APO-U	APO-V	APO-S	APO-U	APO-V	APO-S	APO-U	APO-V	APO-S
1	colon	1192	1117	**879**	1219	1171.7	**885.75**	1278	1266	**895**
2	warpAR10P	1408	1388	**1053**	1470.55	1421.9	**1064.05**	1604	1464	**1088**
3	lymphoma	2037	2338	**1742**	2373.35	2401.6	**1751.6**	2436	2543	**1763**
4	GLIOMA	2170	2466	**1912**	2314.4	2550.55	**1933.8**	2585	2619	**1948**
5	Leukemia_1	3201	3052	**2310**	3276.45	3147.6	**2323.4**	3382	3243	**2341**
6	DLBCL	3235	2833	**2344**	3305.95	3122.75	**2371.2**	3403	3185	**2392**
7	Brain_Tumor_1	3483	3441	**2500**	3535.2	3533.75	**2514.8**	3665	3690	**2531**
8	Prostate_GE	3566	3323	**2542**	3614.25	3508.9	**2593.3**	3675	3615	**2643**
9	ALLAML_M	3843	4180	**3046**	4157.05	4318.1	**3079.85**	4440	4467	**3175**
10	Brain_Tumor_2	5803	5785	**4450**	6187.25	6172.05	**4480.75**	6400	6733	**4498**
11	Prostate_Tumor_1	3546	3450	**2574**	3649	3554.1	**2585.2**	3725	3632	**2617**
12	Leukemia_2	4074	4162	**3109**	4343.15	4363.4	**3158.1**	4759	4722	**3211**

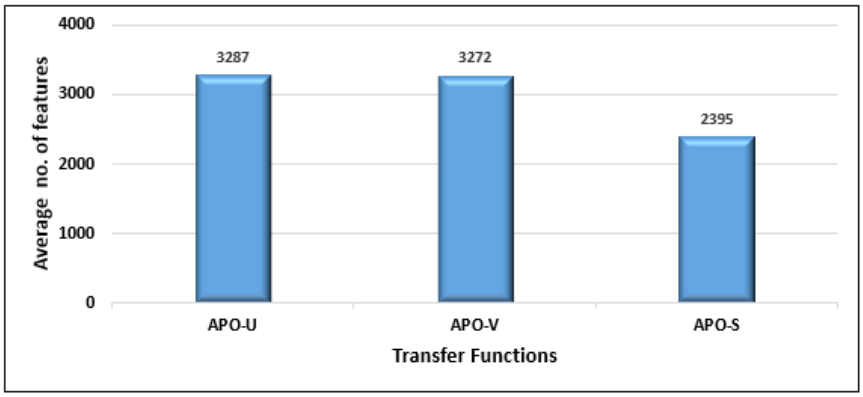


Figure 1. Average fitness, classification accuracy, and number of selected features over all datasets.

- Classification accuracy, evaluated for the best, average, and worst cases
- Number of selected features, assessed for the best, average, and worst scenarios
- Average computational time, used to compare the computational efficiency

Tables 5 and 6 illustrate that the proposed EAPO method surpasses the standard APO in terms of best, average, and worst fitness values, as well as classification accuracy. Specifically, EAPO achieved superior best values across all twelve datasets for both fitness and classification accuracy. In terms of average values, EAPO outperformed APO in 10 datasets, while falling short in 2, namely Leukemia_1 and Brain_Tumor_1. Regarding worst values, the proposed algorithm exhibited better performance in 8 out of the 12 datasets for both fitness and classification accuracy. Furthermore, Fig 2 illustrates the overall average values across all datasets. EAPO demonstrates superior performance with the lowest average fitness value of 0.1222 and the highest average classification accuracy of 0.8795. These results underscore EAPO's effectiveness in maintaining robust performance, avoiding local optima, and consistently achieving high classification accuracy across diverse datasets.

Table 7 presents the number of selected features evaluated for the best, average, and worst scenarios across 20 runs for each algorithm. EAPO consistently outperforms APO across all twelve datasets in terms of all measures of the number of selected features. Additionally, Fig 2 demonstrates the average number of selected features for each algorithm, along with the overall average values across all datasets. Specifically, the proposed method demonstrates superior performance by selecting fewer features, with an average of 1602.58 compared to 3287.13 for the original APO algorithm. This results in a reduction of approximately 51.3%, reflecting a significant decrease in the dimensionality of the datasets. These results underscore EAPO's effectiveness in achieving dimensionality reduction while maintaining robust performance.

Table 5. Comparison of APO and EAPO performance based on best, average, and worst fitness values.

#	Dataset	Best fitness		Average fitness		Worst fitness	
		APO	EAPO	APO	EAPO	APO	EAPO
1	Colon	0.1659	**0.1125**	0.1829	**0.1578**	0.1965	**0.1962**
2	WarpAR10P	0.4249	**0.3895**	0.4299	**0.4138**	0.4328	**0.4307**
3	Lymphoma	0.0566	**0.0417**	0.0575	**0.0503**	0.0576	**0.0556**
4	GLIOMA	0.1435	**0.0996**	0.1438	**0.1012**	0.1444	**0.1194**
5	Leukemia_1	0.0195	**0.0026**	**0.0326**	0.0402	**0.0457**	0.0593
6	DLBCL	0.0580	**0.0008**	0.0667	**0.0334**	0.0828	**0.0544**
7	Brain_Tumor_1	0.1049	**0.0915**	**0.1127**	0.1147	**0.1160**	0.1352
8	Prostate_GE	0.1026	**0.0696**	0.1051	**0.0856**	0.1126	**0.1004**
9	Prostate_Tumor_1	0.1330	**0.0593**	0.1343	**0.0924**	**0.1433**	0.1523
10	ALLAML	0.1300	**0.1110**	0.1408	**0.1335**	**0.1445**	0.1627
11	Leukemia_2	0.1154	**0.0698**	0.1226	**0.0969**	0.1295	**0.1186**
12	Brain_Tumor_2	0.1642	**0.1207**	0.1792	**0.1462**	0.1843	**0.1261**

Table 6. Comparison of APO and EAPO performance according to best, average, and worst classification accuracy.

#	Dataset	Best accuracy		Average accuracy		Worst accuracy	
		APO	EAPO	APO	EAPO	APO	EAPO
1	Colon	0.8385	**0.8872**	0.8214	**0.8435**	**0.8077**	0.8064
2	WarpAR10P	0.5769	**0.6077**	0.5719	**0.586**5	**0.5692**	**0.5692**
3	Lymphoma	0.9479	**0.9584**	0.9479	**0.9511**	**0.9479**	**0.9479**
4	GLIOMA	0.8600	**0.9000**	0.8600	**0.8990**	0.8600	**0.8800**
5	Leukemia_1	0.9867	**1.0000**	**0.9733**	0.9634	**0.9600**	0.9448
6	DLBCL	0.9475	**1.0000**	0.9388	**0.9687**	0.9225	**0.9492**
7	Brain_Tumor_1	0.9000	**0.9111**	**0.8922**	0.8872	**0.8889**	0.8667
8	Prostate_GE	0.9024	**0.9310**	0.9000	**0.9165**	0.8924	**0.9010**
9	Prostate_Tumor_1	0.8719	**0.9410**	0.8705	**0.9089**	0.8614	**0.8824**
10	ALLAML	0.8743	**0.8886**	0.8637	**0.8672**	**0.8600**	0.8486
11	Leukemia_2	0.8895	**0.9314**	0.8823	**0.9059**	0.8752	**0.8771**
12	Brain_Tumor_2	0.8400	**0.8800**	0.8250	**0.8560**	0.8200	**0.8400**

Table 7. Comparison of APO and EAPO performance based on best, average, and worst values for the number of selected features.

#	Dataset	Best number		Average number		Worst number	
		APO	EAPO	APO	EAPO	APO	EAPO
1	Colon	1192	**161**	1219	**571.25**	1278	**1055**
2	WarpAR10P	1408	**267**	1470.55	**1080.25**	1604	**1441**
3	Lymphoma	2037	**225**	2373.35	**738**	2436	**1642**
4	GLIOMA	2170	**280**	2314.4	**523.75**	2585	**977**
5	Leukemia_1	3201	**656**	3276.45	**2103.35**	3382	**3032**
6	DLBCL	3235	**446**	3305.95	**1332.75**	3403	**2393**
7	Brain_Tumor_1	3483	**695**	3535.2	**1777.4**	3665	**2962**
8	Prostate_GE	3566	**368**	3614.25	**1773.1**	3675	**3017**
9	Prostate_Tumor_1	3546	**483**	3649	**1347.85**	3725	**2469**
10	ALLAML	3843	**500**	4157.05	**1482.25**	4440	**3130**
11	Leukemia_2	4074	**983**	4343.15	**2690.05**	4759	**3731**
12	Brain_Tumor_2	5803	**1620**	6187.25	**3810.95**	6400	**6052**

Figure 3 illustrates the overall Improvement Percentage (IP) reflecting the performance of the two algorithms in comparison to the fitness values across the datasets. The following formula is used to calculate the IP:

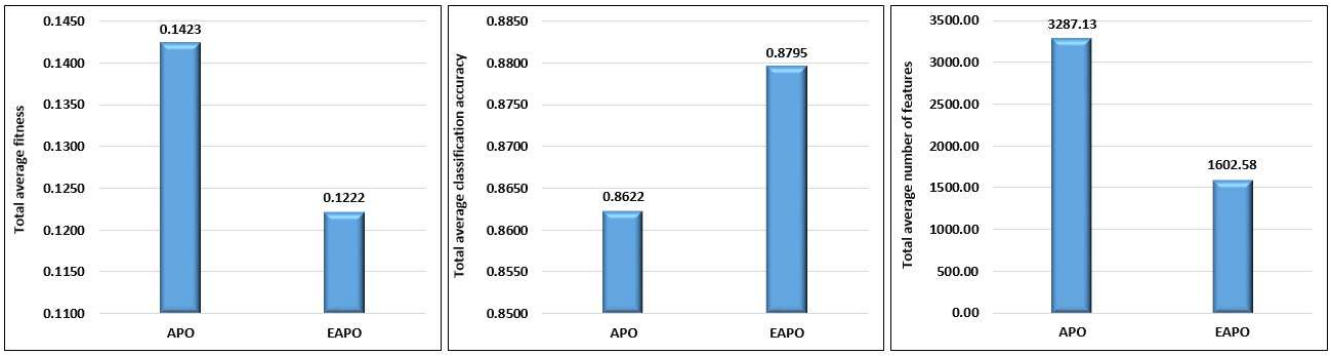


Figure 2. Average fitness, classification accuracy, and number of selected features over all datasets.

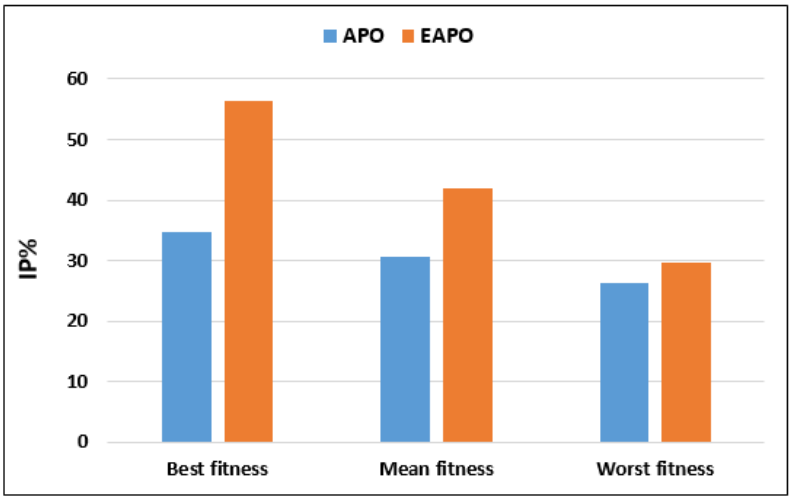


Figure 3. Comparison according to total improvement percentage for the fitness values.

$$\text{Average IP} = \frac{1}{m}\sum_{i=1}^{m}\left(\frac{\text{full}_i - f_{\text{alg}_i}}{\text{full}_i} \times 100\right) \tag{41}$$

where f_{alg} represents the fitness value of both APO and EAPO for the best, mean, and worst cases. The index i ranges from 1 to m, where $m = 12$ represents the total number of datasets. The figure shows that for both algorithms, the overall IP has improved dramatically, showing better performance compared to include all features for every dataset. Furthermore, it is evident that EAPO achieves a much higher IP than APO across all datasets. In Fig 4, the average runtime over 20 executions across all datasets is presented. While our proposed algorithm, EAPO, shows a slightly higher average runtime (82.7 seconds) compared to the original APO (71.8 seconds), this difference is relatively small. Considering the significant improvements achieved by EAPO in fitness values, classification accuracy, and feature selection count, the modest increase of computation time is negligible and well justified.

4.4.3 Third phase: Comparative analysis of the proposed method and other metaheuristic algorithms

The proposed algorithm (EAPO) in this phase is compared with six existing algorithms (BWO, SCA, KOA, COA, SSA, and GWO) and the original APO to verify the results. Various performance measures are employed to assess the algorithms, including mean fitness, classification accuracy, macro precision, macro sensitivity, and the number of selected features. Tables 8 and 9 present the outcomes achieved by the eight algorithms, focusing on the mean fitness values and classification accuracy, respectively. From studying the results, EAPO demonstrates lower fitness values and higher classification accuracy compared to the other algorithms across all twelve datasets. Additionally, Figs 5 and 6 illustrate the total average fitness and total classification accuracy, respectively, over all the datasets. One may note that EAPO achieves the minimum average fitness value of 0.1222 and the highest classification accuracy of 0.8795.

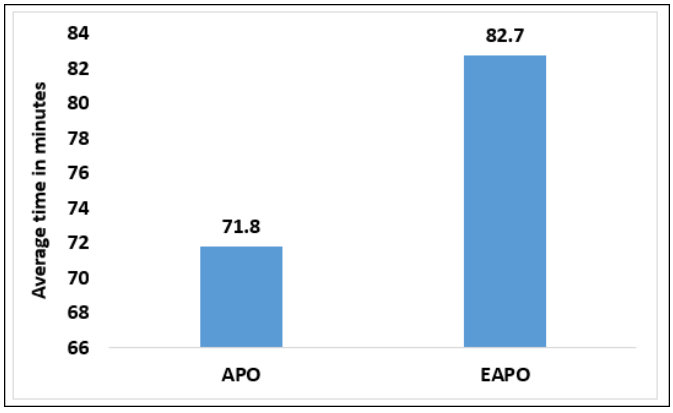


Figure 4. Average computational time across all the datasets.

Moreover, Table 10 displays the average number of selected features in each dataset. One may observe that the proposed algorithm outperformed in 9 out of 12 datasets. Additionally, Fig 7 shows the average number of selected features across all the datasets, with EAPO ranking first at 1602.6 features. The findings from these experiments indicate that the integration of the U-shaped function considerably improves both the robustness and performance of the proposed algorithm. Furthermore, using the local escaping operator improves the exploitation capabilities of the EAPO algorithm. The adoption of k-fold cross-validation proves to be an effective strategy to mitigate the overfitting issue. Additionally, the kNN classifier, as a wrapper method, demonstrates its ability to provide high-performing solutions and robust learning from the training set.

Table 8. Methods compared in terms of their average fitness values.

#	Dataset	BWO	SCA	KOA	COA	SSA	GWO	EAPO	APO
1	Colon	0.1751	0.1687	0.2437	0.2226	0.1881	0.1935	**0.1578**	0.1829
2	WarpAR10P	0.4452	0.4791	0.4820	0.4271	0.4488	0.4562	**0.4138**	0.4299
3	Lymphoma	0.0706	0.0777	0.0887	0.0558	0.0756	0.0821	**0.0503**	0.0575
4	GLIOMA	0.1271	0.1305	0.1931	0.1363	0.1419	0.1435	**0.1012**	0.1438
5	Leukemia_1	0.0509	0.0595	0.0960	0.0460	0.0668	0.0668	**0.0402**	0.0326
6	DLBCL	0.0595	0.0749	0.0705	0.0528	0.0650	0.0665	**0.0334**	0.0667
7	Brain_Tumor_1	0.1166	0.1328	0.1303	0.1303	0.1188	0.1218	**0.1147**	0.1127
8	Prostate_GE	0.1026	0.1346	0.1256	0.0902	0.0990	0.1118	**0.0856**	0.1051
9	Prostate_Tumor_1	0.1141	0.1087	0.1201	0.1274	0.1229	0.1090	**0.0924**	0.1343
10	ALLAML	0.1479	0.1411	0.1387	0.1349	0.1417	0.1461	**0.1335**	0.1408
11	Leukemia_2	0.1265	0.1334	0.1357	0.1224	0.1176	0.1165	**0.0969**	0.1226
12	Brain_Tumor_2	0.1782	0.1664	0.2317	0.1636	0.1619	0.1718	**0.1462**	0.1792

In the context of feature selection and classification, macro precision and macro sensitivity are critical measures for evaluating the effectiveness of algorithms. Precision reflects the algorithm's ability to make accurate positive predictions. Sensitivity (recall), on the other hand, evaluates its ability to correctly identify positive instances. Across 12 datasets, the proposed feature selection metaheuristic algorithm outperformed competing approaches in both macro precision and macro sensitivity in 10 out of 12 datasets, as shown in Table 11 and Table 12. This improvement highlights the algorithm's balanced performance, excelling not only in identifying relevant features that contribute to accurate positive predictions but also in excluding irrelevant ones, thereby reducing false positives. Such advancements underscore the algorithm's robustness and consistency in delivering high-quality feature subsets, which leads to improved classification accuracy and reliability in decision-making processes. This dual superiority demonstrates the algorithm's capability to maintain a comprehensive understanding of the dataset's structure, ensuring effective exploration and exploitation in the search space.

Moreover, as illustrated in Fig 8 and Fig 9, EAPO consistently outperformed all other algorithms in terms of convergence in most of the 12 datasets. This detailed comparison underscores EAPO's superior performance. The convergence curves in these figures further highlight EAPO's ability to address premature convergence by maintaining a balanced trade-off between exploitation and exploration, which enhances population diversity. Figs 10 and 11 present the boxplots for the 12 datasets, visually representing the algorithms' average performance. Each boxplot illustrates five key elements: the minimum, maximum,

Table 9. Methods compared in terms of their classification accuracy.

#	Dataset	BWO	SCA	KOA	COA	SSA	GWO	EAPO	APO
1	Colon	0.8251	0.8366	0.7598	0.7759	0.8157	0.8110	**0.8435**	0.8214
2	WarpAR10P	0.5515	0.5227	0.5192	0.5719	0.5523	0.5458	**0.586**5	0.5719
3	Lymphoma	0.9326	0.9279	0.9163	0.9464	0.9289	0.9234	**0.9511**	0.9479
4	GLIOMA	0.8740	0.8750	0.8110	0.8660	0.8620	0.8610	**0.8990**	0.8600
5	Leukemia_1	0.9531	0.9469	0.9092	0.9565	0.9381	0.9392	**0.9634**	0.9733
6	DLBCL	0.9439	0.9314	0.9348	0.9498	0.9399	0.9391	**0.968**7	0.9388
7	Brain_Tumor_1	0.8856	0.8722	0.8744	0.8733	0.8856	0.8833	**0.887**2	0.8922
8	Prostate_GE	0.9024	0.8709	0.8792	0.9110	0.9056	0.8933	**0.9165**	0.9000
9	Prostate_Tumor_1	0.8884	0.8966	0.8847	0.8753	0.8814	0.8964	**0.9089**	0.8705
10	ALLAML	0.8524	0.8640	0.8659	0.8649	0.8624	0.8587	**0.867**2	0.8637
11	Leukemia_2	0.8796	0.8720	0.8690	0.8821	0.8868	0.8888	**0.9059**	0.8823
12	Brain_Tumor_2	0.8240	0.8390	0.7720	0.8380	0.8420	0.8330	**0.8560**	0.8250

Table 10. Methods compared in terms of the average number of selected features.

#	Dataset	BWO	SCA	KOA	COA	SSA	GWO	EAPO	APO
1	Colon	**373.65**	1381.8	1180.45	138.75	1122.85	1274.4	571.25	1219
2	WarpAR10P	**304.95**	1581.95	1440.5	800.7	1329.8	1574	1080.25	1470.55
3	Lymphoma	1561.35	2532	2337.1	1098.75	2122.75	2543	**738**	2373.35
4	GLIOMA	1028.9	2998.5	2650.35	1620.65	2335.05	2618.45	**523.7**5	2314.4
5	Leukemia_1	2376.25	3677.3	3230.7	1546.15	2932.95	3520.2	**2103.35**	3276.45
6	DLBCL	2158.65	3823.5	3275.65	1672.5	2978.15	3394.9	**1332.7**5	3305.95
7	Brain_Tumor_1	1963.65	3728	3546.35	2880.85	3234.75	3720.55	**1777.4**	3535.2
8	Prostate_GE	3565.4	4049.7	3563.3	1245.4	3296.3	3720.45	**1773.1**	3614.25
9	Prostate_Tumor_1	2202.55	3828.25	3581.65	2391.25	3277.25	3836.55	**1347.85**	3649
10	ALLAML	**1236.9**	4583.55	4252.65	774.15	3871.4	4453	1482.25	4157.05
11	Leukemia_2	5201.6	4813.2	4282.8	4063.3	3887.6	4519.05	**2690.05**	4343.15
12	Brain_Tumor_2	4134.9	7238.55	6216.1	3331.5	5645.8	6706.7	**3810.95**	6187.25

Table 11. Macro precision of the proposed algorithm compared to other algorithms.

#	Dataset	BWO	SCA	KOA	COA	SSA	GWO	EAPO	APO
1	Colon	0.8391	0.8413	0.7651	0.7813	0.8211	0.8087	**0.8513**	0.8279
2	WarpAR10P	0.5952	0.5949	0.5307	0.6561	0.5289	0.5961	**0.6450**	0.6284
3	Lymphoma	0.8325	0.8390	0.7863	0.8461	0.8309	0.8189	**0.8608**	0.8530
4	GLIOMA	0.9067	0.9104	0.8113	0.8944	0.8600	0.8725	**0.9302**	0.8958
5	Leukemia_1	0.9244	0.9665	0.9191	0.9657	0.9111	0.9665	0.9794	**0.9862**
6	DLBCL	0.9265	0.9026	0.9083	0.9467	0.9212	0.9028	**0.9562**	0.9072
7	Brain_Tumor_1	0.7115	0.6962	0.6383	0.6650	0.6779	0.7072	0.6610	**0.7393**
8	Prostate_GE	0.9081	0.8737	0.8880	0.9149	0.9073	0.8960	**0.9266**	0.9121
9	ALLAML	0.8998	0.8971	0.9019	0.9142	0.8971	0.8998	**0.9193**	0.8989
10	Brain_Tumor_2	0.7968	0.8321	0.8179	0.8260	0.8665	0.8612	**0.8674**	0.8165
11	Prostate_Tumor_1	0.8915	0.9067	0.8926	0.8799	0.8856	0.9034	**0.9134**	0.8774
12	Leukemia_2	0.6883	0.7629	0.6968	0.7423	0.7452	0.7505	**0.7966**	0.7491

median, first quartile, and third quartile of the data. The median value is indicated by the line inside the box. Classification accuracy is reflected in these boxplots, which are derived from 20 independent runs of each algorithm. EAPO exhibits higher boxplots than other algorithms, indicating its superior performance. Moreover, the median values of EAPO are consistently greater than the median values of the competing algorithms, reaffirming its robustness. Overall, these boxplots demonstrate that EAPO is not only competitive but also often superior across the majority of datasets.

The stability of EAPO's performance stems from its well-designed transformation function and the incorporation of the

Table 12. Macro sensitivity of the proposed method compared to other methods.

#	Dataset	BWO	SCA	KOA	COA	SSA	GWO	EAPO	APO
1	Colon	0.8038	0.8095	0.7201	0.7386	0.8005	0.7844	**0.8283**	0.8075
2	WarpAR10P	0.5435	0.5287	0.5237	0.5685	0.5393	0.5400	**0.5932**	0.5800
3	Lymphoma	0.8473	0.8535	0.8254	0.8683	0.8510	0.8367	**0.8800**	0.8714
4	GLIOMA	0.8763	0.8788	0.7933	0.8804	0.8567	0.8679	**0.9058**	0.8750
5	Leukemia_1	0.9017	0.9143	0.8523	0.9328	0.9013	0.9110	**0.9596**	0.9546
6	DLBCL	0.9604	0.9545	0.9488	0.9633	0.9589	0.9595	**0.9790**	0.9576
7	Brain_Tumor_1	0.6884	0.6630	0.6590	0.6571	0.6805	0.6850	0.6908	**0.6955**
8	Prostate_GE	0.9027	0.8711	0.8787	0.9119	0.9060	0.8934	**0.9164**	0.9004
9	ALLAML	0.7961	0.8213	0.8169	0.8063	0.8065	0.8087	0.8080	**0.8109**
10	Brain_Tumor_2	0.7979	0.8158	0.7871	0.8192	0.8496	0.8367	**0.8358**	0.8208
11	Prostate_Tumor_1	0.8886	0.8964	0.8856	0.8749	0.8814	0.8963	**0.9088**	0.8709
12	Leukemia_2	0.7133	0.7092	0.6743	0.7204	0.7228	0.7319	**0.7767**	0.7314

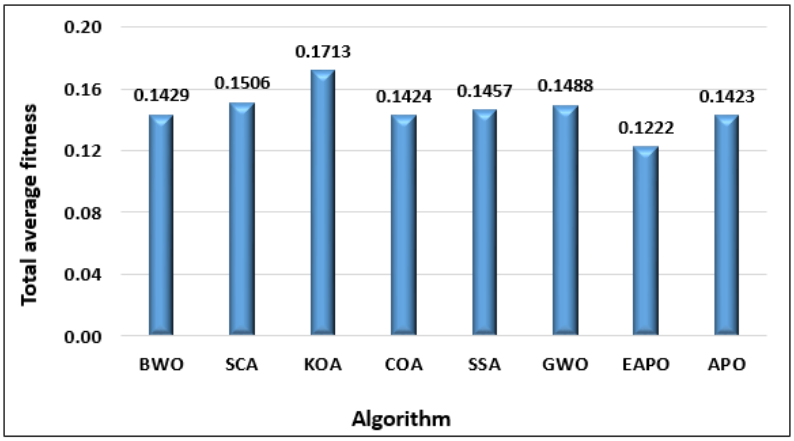


Figure 5. The total average fitness values for each algorithm.

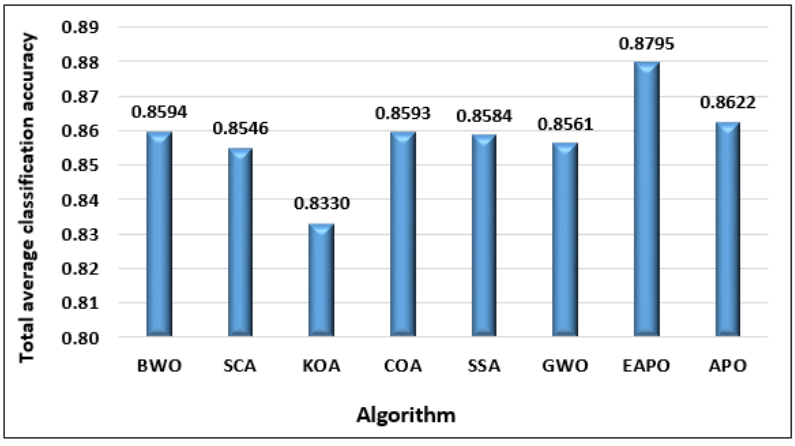


Figure 6. The total average classification accuracy for each algorithm.

local escaping operator into the original APO algorithm. These enhancements allow EAPO to maintain solution diversity more effectively than other algorithms, ensuring robust performance across different scenarios. Moreover, as shown in Tables 8, the fitness values achieved by EAPO consistently surpass those of competing algorithms, highlighting its ability to avoid local optima a challenge that often hinders the performance of other approaches. Additionally, as shown in Table 10, EAPO exhibits remarkable exploration capabilities, as evidenced by its selection of fewer features in most of the 12 datasets. This highlights

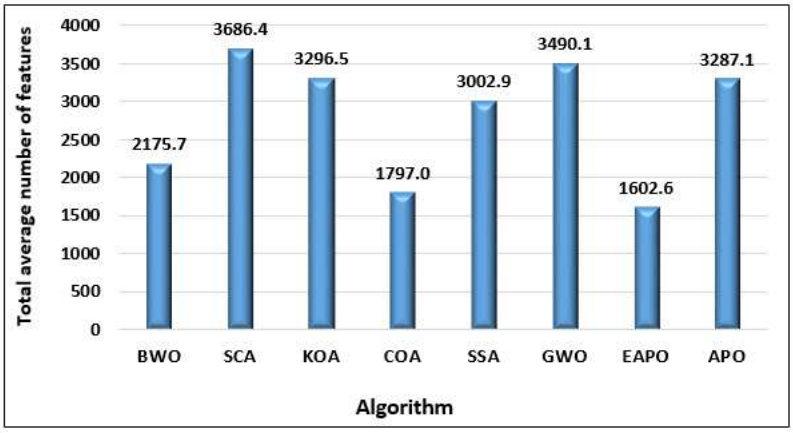


Figure 7. The average number of selected features by each algorithm.

its effectiveness in achieving a balance between exploration and exploitation. This exploration advantage is further reinforced by the higher classification accuracy achieved by EAPO in most datasets. These findings position EAPO as a leader among other competing optimization algorithms, excelling in feature selection and overall performance.

To test if two sample means show statistically notable differences, the Wilcoxon signed-rank test is employed in this study. Table 13 presents the results of the Wilcoxon signed-rank test for six randomly selected datasets (warpAR10P, lymphoma, Prostate GE, Prostate_Tumor_1, ALLAML, and Leukemia_2).

The column **No. R+** indicates the number of positive ranks where the proposed method outperforms the compared algorithms, which include BWO, SCA, KOA, COA, SSA, GWO, and the original version. Conversely, **No. R-** indicates the number of negative ranks where the proposed method falls short compared to the other algorithms. The **No. ties** column refers to the number of instances where both algorithms achieve equal ranks.

The values **Sum R+** and **Sum R-** represent the sum of positive and negative ranks, respectively. For the selected datasets, it is evident that **Sum R+** consistently exceeds **Sum R-**, indicating that EAPO demonstrates significant improvement over other algorithms across most datasets.

With a significance level of $\alpha = 0.05$, the Wilcoxon signed-rank test confirms that there is a statistically significant variation in performance between the proposed method and the compared methods, as the p-value is consistently less than α. In Table 13, p-values greater than 0.05 are highlighted.

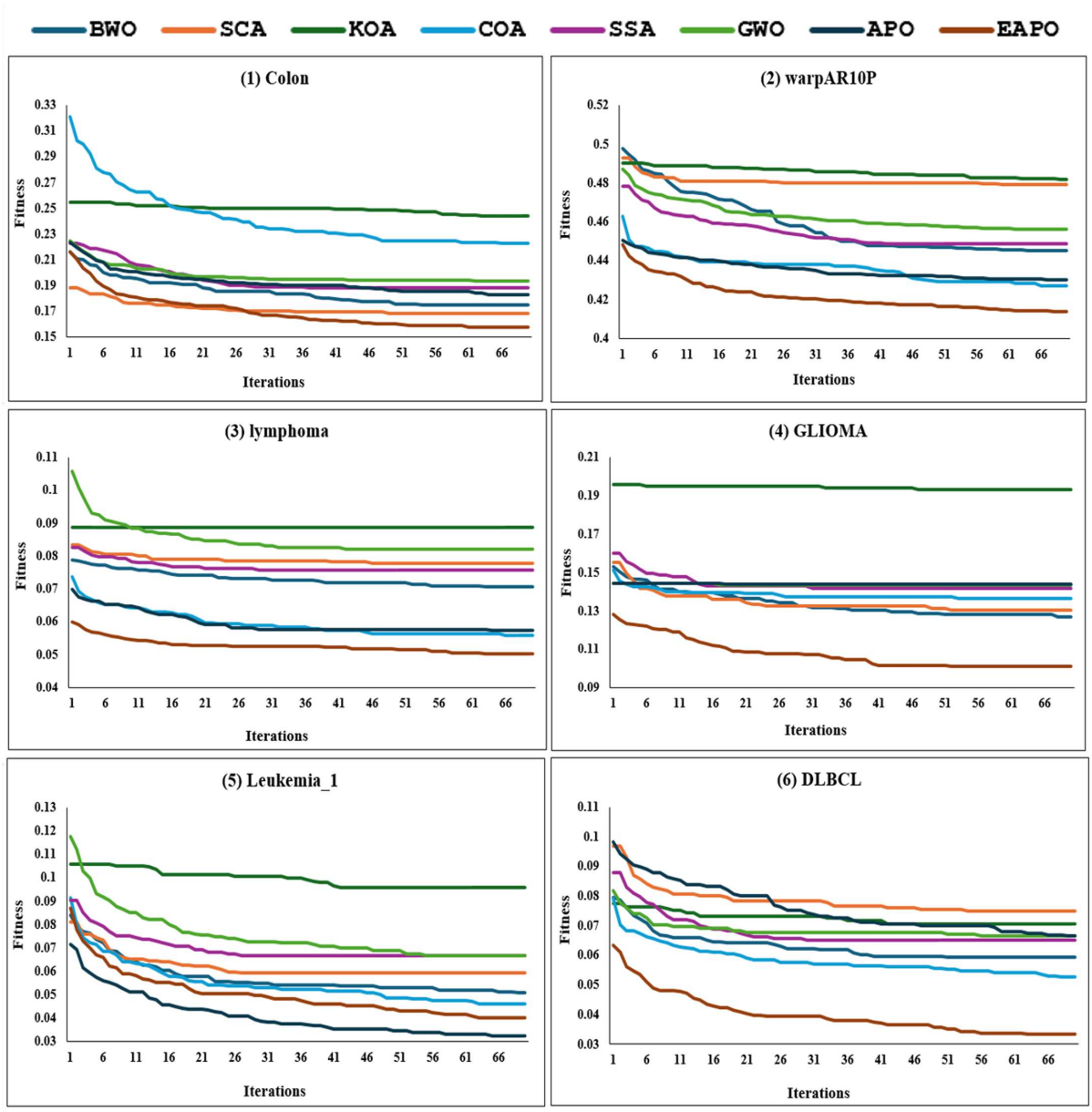


Figure 8. Comparison of the convergence of EAPO with compared metaheuristics for datasets 1 through 6 in Table 1.

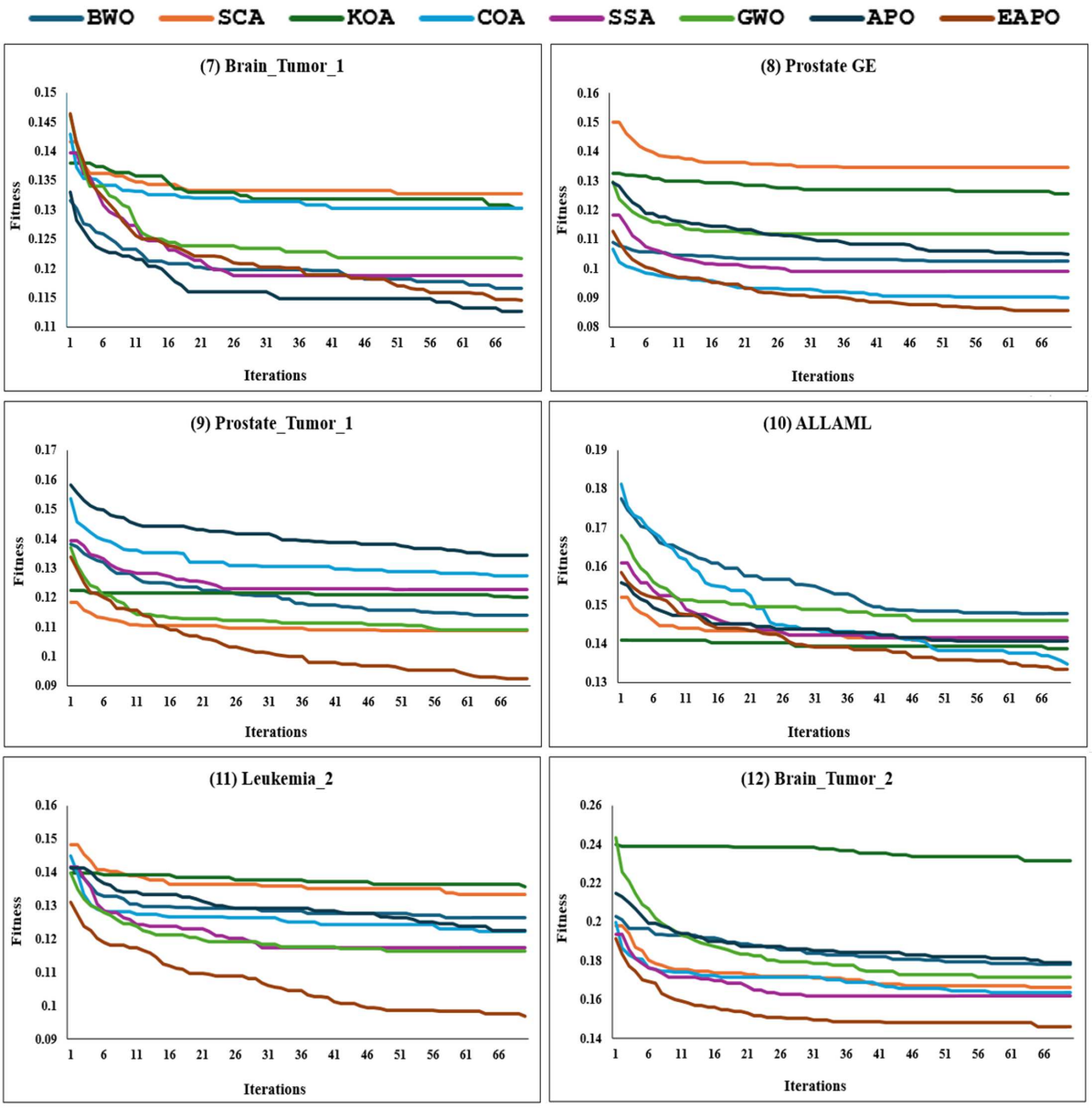


Figure 9. Comparison of the convergence of EAPO with compared metaheuristics for datasets 7 through 12 in Table 1.

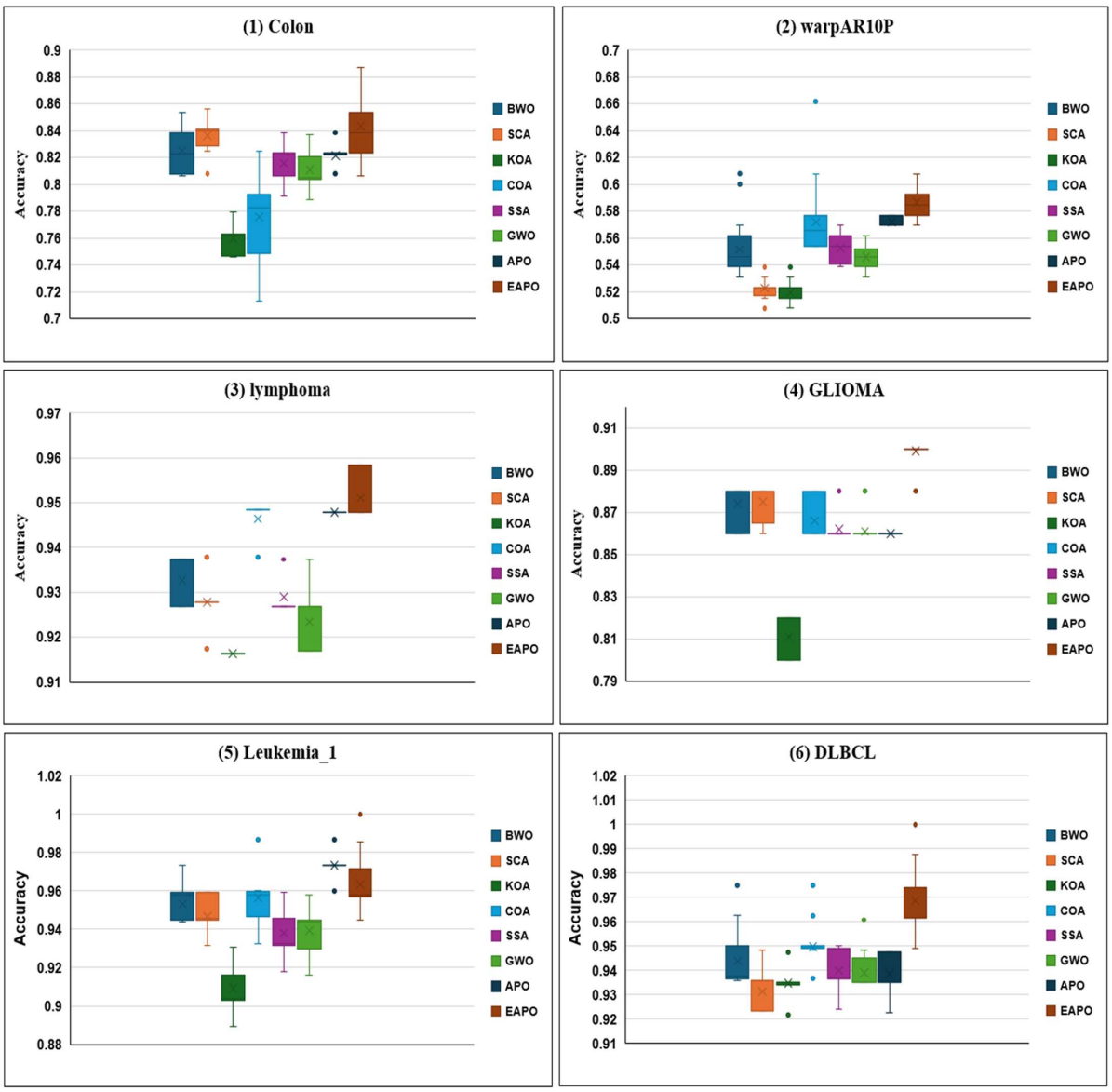


Figure 10. Comparison of the boxplots of EAPO with compared metaheuristics for datasets 1 through 6 in Table 1.

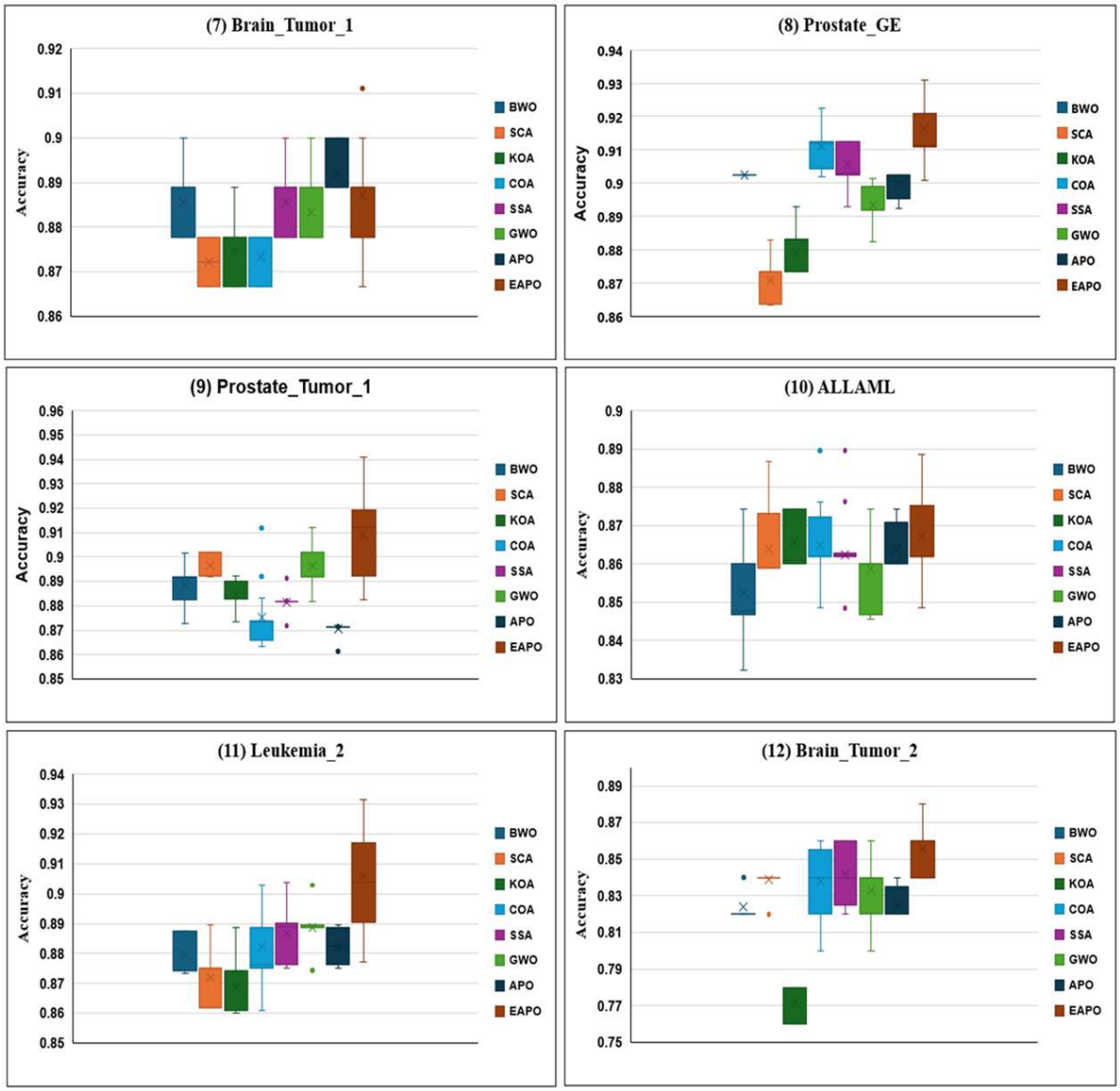


Figure 11. Comparison of the boxplots of EAPO with compared metaheuristics for datasets 7 through 12 in Table 1.

Table 13. Results of the Wilcoxon signed ranks test.

Dataset	Algorithm	No. R+	No. R-	No. ties	Sum_R+	Sum_R-	p-value
WarpAR10P	BWO	18	2	0	202	8	0.001
	SCA	20	0	0	210	0	0.001
	KOA	20	0	0	210	0	0.001
	COA	16	4	0	171	39	0.014
	SSA	20	0	0	210	0	0.001
	GWO	20	0	0	210	0	0.001
	APO	19	1	0	207	3	0.001
Lymphoma	BWO	20	0	0	210	0	0.001
	SCA	20	0	0	210	0	0.001
	KOA	20	0	0	210	0	0.001
	COA	15	5	0	181	29	0.005
	SSA	20	0	0	210	0	0.001
	GWO	20	0	0	210	0	0.001
	APO	20	0	0	210	0	0.001
Prostate GE	BWO	20	0	0	210	0	0.001
	SCA	20	0	0	210	0	0.001
	KOA	20	0	0	210	0	0.001
	COA	12	8	0	149	61	0.1
	SSA	19	1	0	208	2	0.001
	GWO	20	0	0	210	0	0.001
	APO	20	0	0	210	0	0.001
Prostate_Tumor_1	BWO	20	0	0	210	0	0.001
	SCA	17	3	0	189	21	0.002
	KOA	20	0	0	210	0	0.001
	COA	19	1	0	208	2	0.001
	SSA	20	0	0	210	0	0.001
	GWO	18	2	0	197	13	0.001
	APO	20	0	0	210	0	0.001
ALLAML	BWO	16	4	0	190	20	0.002
	SCA	16	4	0	168	42	0.019
	KOA	15	5	0	141	69	0.179
	COA	11	9	0	118	92	0.627
	SSA	16	4	0	168	42	0.019
	GWO	17	3	0	177	33	0.007
	APO	16	4	0	160	50	0.040
Leukemia_2	BWO	19	1	0	205	5	0.001
	SCA	20	0	0	210	0	0.001
	KOA	20	0	0	210	0	0.001
	COA	20	0	0	210	0	0.001
	SSA	19	1	0	203	7	0.001
	GWO	19	1	0	203	7	0.001
	APO	20	0	0	210	0	0.001

5 Conclusion

Feature selection is a critical step in classification tasks, as it helps reduce time complexity, makes the learning model simple, and improves the overall performance of the classifiers. In this article, the EAPO algorithm is proposed for feature selection. To enable the binary adaptation, three categories of TFs, S-shaped, V-shaped, and U-shaped, are evaluated in order to determine the most effective mapping from the continuous search space to the binary domain. Furthermore, the method incorporates local escaping operator operations into the search process of the APO algorithm. This mechanism improves the quality of the solution by updating the positions based on specific criteria, thereby preventing APO from becoming trapped in local optima and improving convergence performance.

The performance of the proposed method was compared with several well-known wrapper-based feature selection algorithms, including BOA, SCA, KOA, COA, SSA, and GWO. Experimental results were analyzed from multiple perspectives, including mean fitness, classification accuracy, macro precision, macro sensitivity, and the number of selected features. The outcomes obtained from experiments conducted on high-dimensional datasets demonstrated that the proposed method effectively eliminated redundant and irrelevant features. The integration of an appropriate TF and the local escaping operator mechanism led to classification results that outperformed those achieved by existing wrapper-based feature selection methods. Additionally, the Wilcoxon test was employed to show whether the variations between the proposed method and the compared algorithms are statistically significant. The statistical tests confirmed that the differences were statistically significant. For future work, hybridization approaches could also be explored to improve the algorithm's exploration capabilities. Moreover, integrating quantum computing concepts into the proposed algorithm is expected to yield promising results.

Funding

This research received no external funding.

Data availability

Datasets used in this study are publicly available at https://ckzixf.github.io/dataset.html and https://jundongl.github.io/scikit-feature/datasets.html.

Author contributions

Conceptualization: M.S., and D.A.E.; methodology: M.S., A.F., and D.A.E.; software: M.S., and D.A.E.; investigation: M.S. and D.A.E.; Formal analysis: M.S., A.F., and D.A.E.; Writing—original draft: M.S., A.F., and D.A.E.; Writing—review and editing: M.S., A.F., and D.A.E. All authors have reviewed the manuscript.

Declarations

Competing interests
The authors declare no competing interests.

Consent for publication
This article does not contain any studies with human participants or animals performed by any of the authors.

Ethics approval and consent to participate
T he results/data/figures in this manuscript have not been published elsewhere, nor are they under consideration by another publisher. All the material is owned by the authors, and/or no permissions are required.

References

1. Nguyen, B. H., Xue, B. & Zhang, M. A survey on swarm intelligence approaches to feature selection in data mining. *Swarm and Evolutionary Computation* **54**, 100663, DOI: https://doi.org/10.1016/j.swevo.2020.100663 (2020).
2. Hua, J., Tembe, W. D. & Dougherty, E. R. Performance of feature-selection methods in the classification of high-dimension data. *Pattern Recognition* **42**, 409–424, DOI: https://doi.org/10.1016/j.patcog.2008.08.001 (2009).
3. Dash, M. & Liu, H. Feature selection for classification. *Intelligent Data Analysis* **1**, 131–156, DOI: https://doi.org/10.3233/ida-1997-1302 (1997).
4. Sun, Z. Parallel feature selection based on MapReduce. In *Computer Engineering and Networking: Proceedings of the 2013 International Conference on Computer Engineering and Network (CENet2013)*. Cham: Springer International Publishing 299–306, DOI: https://doi.org/10.1007/978-3-319-01766-2_35 (2014).
5. Liu, H. & Motoda, H. Feature selection for knowledge discovery and data mining, vol. 454 (*Springer science & business media*, 2012).
6. Abdelhamid, A. A. *et al.* Innovative feature selection method based on hybrid sine cosine and dipper throated optimization algorithms. *IEEE Access* **11**, 79750–79776, DOI: https://doi.org/10.1109/access.2023.3298955 (2023).
7. Li, Y., Li, T. & Liu, H. Recent advances in feature selection and its applications. *Knowledge and Information Systems* **53**, 551–577, DOI: https://doi.org/10.1007/s10115-017-1059-8 (2017).
8. Guyon, I. & Elisseeff, A. An introduction to variable and feature selection. *Journal of Machine Learning Research* **3**, 1157–1182 (2003).
9. Mohammed, I. J., Al-Nuaimi, B. T., Baker, T. I. *et al*. Weather forecasting over Iraq using machine learning. *Journal of Artificial Intelligence and Metaheuristics* **2**, 39–45, DOI: https://doi.org/10.54216/jaim.020204 (2022).
10. Žerovnik, J. Heuristics for np-hard optimization problems-simpler is better!? *Logistics, Supply Chain, Sustainability and global challenges* **6**, 1–10, DOI: https://doi.org/10.1515/jlst-2015-0006 (2015).
11. Hammouri, A. I., Mafarja, M., Al-Betar, M. A., Awadallah, M. A. & Abu-Doush, I. An improved dragonfly algorithm for feature selection. *Knowledge-Based Systems* **203**, 106131, DOI: https://doi.org/10.1016/j.knosys.2020.106131 (2020).
12. Ahmed, S., Sheikh, K. H., Mirjalili, S. & Sarkar, R. Binary simulated normal distribution optimizer for feature selection: Theory and application in covid-19 datasets. *Expert Systems with Applications* **200**, 116834, DOI: https://doi.org/10.1016/j.eswa.2022.116834 (2022).

13. Emary, E. & Zawbaa, H. M. Feature selection via lèvy antlion optimization. *Pattern Analysis and Applications* **22**, 857–876, DOI: https://doi.org/10.1007/s10044-018-0695-2 (2019).

14. Ji, B. *et al*. Bio-inspired feature selection: An improved binary particle swarm optimization approach. *IEEE Access* **8**, 85989–86002, DOI: https://doi.org/10.1109/access.2020.2992752 (2020).

15. Wang, W.-c., Tian, W.-c., Xu, D.-m. & Zang, H.-f. Arctic puffin optimization: A bio-inspired metaheuristic algorithm for solving engineering design optimization. *Advanced Engineering Software* **195**, 103694, DOI: https://doi.org/10.1016/j.advengsoft.2024.103694 (2024).

16. Abdollahzadeh, B., Gharehchopogh, F. S.& Mirjalili, S. African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems. *Computers & Industrial Engineering* **158**, 107408, DOI: https://doi.org/10.1016/j.cie.2021.107408 (2021).

17. Li, S., Chen, H.,Wang, M., Heidari, A. A. & Mirjalili, S. Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems* **111**, 300–323, DOI: https://doi.org/10.1016/j.future.2020.03.055 (2020).

18. Abdollahzadeh, B., Soleimanian Gharehchopogh, F. & Mirjalili, S. Artificial gorilla troops optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems. *International Journal of Intelligent Systems* **36**, 5887–5958, DOI: https://doi.org/10.1002/int.22535 (2021).

19. Abualigah, L., Abd Elaziz, M., Sumari, P., Geem, Z.W. & Gandomi, A. H. Reptile search algorithm (rsa): A nature-inspired meta-heuristic optimizer. *Expert Systems with Applications* **191**, 116158, DOI: https://doi.org/10.1016/j.eswa.2021.116158 (2022).

20. Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M. & Gandomi, A. H. The arithmetic optimization algorithm. *Computational Methods in Applied Mechanics and Engineering* **376**, 113609, DOI: https://doi.org/10.1016/j.cma.2020.113609 (2021).

21. Hansen, N., Müller, S. D. & Koumoutsakos, P. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary Computation* **11**, 1–18, DOI: https://doi.org/10.1162/106365603321828970 (2003).

22. Ahmadianfar, I., Heidari, A. A., Gandomi, A. H., Chu, X. & Chen, H. Run beyond the metaphor: An efficient optimization algorithm based on runge kutta method. *Expert Systems with Applications* **181**, 115079, DOI: https://doi.org/10.1016/j.eswa.2021.115079 (2021).

23. Zhong, C., Li, G. & Meng, Z. Beluga whale optimization: A novel nature inspired metaheuristic algorithm. *Knowledge-Based Systems* **251**, 109215, DOI: https://doi.org/10.1016/j.knosys.2022.109215 (2022).

24. Heidari, A. A. *et al.* Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems* **97**, 849–872, DOI: https://doi.org/10.1016/j.future.2019.02.028 (2019).

25. Tubishat, M., Abushariah, M. A., Idris, N. & Aljarah, I. Improved whale optimization algorithm for feature selection in arabic sentiment analysis. *Applied Intelligence* **49**, 1688–1707, DOI: https://doi.org/10.1007/s10489-018-1334-8 (2019).

26. Zhang, Y., Gong, D.-w., Gao, X.-z., Tian, T. & Sun, X.-y. Binary differential evolution with self-learning for multi-objective feature selection. *Information Sciences* **507**, 67–85, DOI: https://doi.org/10.1016/j.ins.2019.08.040 (2020).

27. Lenin, K., Reddy, B. R. & Suryakalavathi, M. Hybrid tabu search-simulated annealing method to solve optimal reactive power problem. *International Journal of Electrical Power & Energy Systems* **82**, 87–91, DOI: https://doi.org/10.1016/j.ijepes.2016.03.007 (2016).

28. Wu, D. *et al.* Enhance teaching-learning-based optimization for tsallis-entropy-based feature selection classification approach. *Processes* **10**, 360, DOI: https://doi.org/10.3390/pr10020360 (2022).

29. Zhang, K., Liu, Y., Mei, F., Sun, G. & Jin, J. Ibgjo: improved binary golden jackal optimization with chaotic tent map and cosine similarity for feature selection. *Entropy* **25**, 1128, DOI: https://doi.org/10.3390/e25081128 (2023).

30. Holland, J. H. Genetic algorithms. *Scientific American* **267**, 66–73, DOI: https://doi.org/10.1038/scientificamerican0792-66 (1992).

31. Storn, R. & Price, K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization* **11**, 341–359, DOI: https://doi.org/10.1023/a:1008202821328 (1997).

32. Rostami, M., Berahmand, K. & Forouzandeh, S. A novel community detection based genetic algorithm for feature selection. *Journal of Big Data* **8**, 2, DOI: https://doi.org/10.1186/s40537-020-00398-3 (2021).

33. Oh, I.-S., Lee, J.-S. & Moon, B.-R. Hybrid genetic algorithms for feature selection. *EEE Transactions on Pattern Analysis and Machine Intelligence* **26**, 1424–1437, DOI: https://doi.org/10.1109/tpami.2004.105 (2004).

34. Dey, A. *et al.* A hybrid meta-heuristic feature selection method using golden ratio and equilibrium optimization algorithms for speech emotion recognition. *IEEE Access* **8**, 200953–200970, DOI: https://doi.org/10.1109/access.2020.3035531 (2020).

35. Mafarja, M. & Abdullah, S. Investigating memetic algorithm in solving rough set attribute reduction. *International Journal of Computer Applications in Technology.* **48**, 195–202, DOI: https://doi.org/10.1504/ijcat.2013.056915 (2013).

36. Rao, R. V., Savsani, V. J. & Vakharia, D. P. Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. *Computer-Aided Design* **43**, 303–315, DOI: https://doi.org/10.1016/j.cad.2010.12.015 (2011).

37. Ali, M. A., PP, F. R. & Abd Elminaam, D. S. An efficient heap based optimizer algorithm for feature selection. *Mathematics* **10**, 2396, DOI: https://doi.org/10.3390/math10142396 (2022).

38. Eberhart, R. & Kennedy, J. Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Networks,* vol. 4, 1942–1948, DOI: https://doi.org/10.1109/icnn.1995.488968 (Citeseer, 1995).

39. Mirjalili, S., Mirjalili, S. M. & Lewis, A. Grey wolf optimizer. *Advanced Engineering Software* **69**, 46–61, DOI: https://doi.org/10.1016/j.advengsoft.2013.12.007 (2014).

40. Pereira, L. *et al.* A binary cuckoo search and its application for feature selection.*In: Cuckoo search and firefly algorithm: theory and applications.* Cham: Springer International Publishing, 141–154, DOI: https://doi.org/10.1007/978-3-319-02141-6_7 (2013).

41. Wen, L., Yin, Q. & Guo, P. Ant colony optimization algorithm for feature selection and classification of multispectral remote sensing image. In *IGARSS 2008-2008 IEEE International Geoscience and Remote Sensing Symposium,* vol. 2, II–923, DOI: https://doi.org/10.1109/igarss.2008.4779146 (IEEE, 2008).

42. Alzubi, O. A., Alzubi, J. A., Al-Zoubi, A. M., Hassonah, M. A. & Kose, U. An efficient malware detection approach with feature weighting based on harris hawks optimization. *Cluster Computing.* 1–19, DOI: https://doi.org/10.1007/s10586-021-03459-1 (2022).

43. Wan, Y., Wang, M., Ye, Z. & Lai, X. A feature selection method based on modified binary coded ant colony optimization algorithm. *Applied Soft Computing* **49**, 248–258, DOI: https://doi.org/10.1016/j.asoc.2016.08.011 (2016).

44. Jona, J.& Nagaveni, N. Ant-cuckoo colony optimization for feature selection in digital mammogram. *Pakistan Journal of Biological Sciences* **17**, 266–271, DOI: https://doi.org/10.3923/pjbs.2014.266.271 (2014).

45. Moradi, P.& Gholampour, M. A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy. *Applied Soft Computing* **43**, 117–130, DOI: https://doi.org/10.1016/j.asoc.2016.01.044 (2016).

46. Mao, B. *et al.* A hybrid differential evolution and particle swarm optimization algorithm for numerical kinematics solution of remote maintenance manipulators. *Fusion Engineering and Design* **124**, 587–590, DOI: https://doi.org/10.1016/j.fusengdes.2017.03.042 (2017).

47. Shahin, I. *et al.* An efficient feature selection method for arabic and english speech emotion recognition using grey wolf optimizer. *Applied Acoustics* **205**, 109279, DOI: https://doi.org/10.1016/j.apacoust.2023.109279 (2023).

48. Esfandiari, A., Farivar, F.& Khaloozadeh, H. Fractional-order binary bat algorithm for feature selection on high-dimensional microarray data. *Journal of Ambient Intelligence and Humanized Computing* **14**, 7453–7467, DOI: https://doi.org/10.1007/s12652-022-04450-3 (2023).

49. Özbay, F. A. An enhanced zebra optimization algorithm with multiple strategies for global optimization and feature selection problems: A hepatocellular carcinoma case study. *IEEE Access* DOI: https://doi.org/10.1109/access.2025.3541975 (2025).

50. Xiao, J., Xu, W. & Chen, J. Social media emotional state classification prediction based on arctic puffin algorithm (apo) optimization of transformer mode. In *2024 International Conference on Electrical, Communication and Computer Engineering (ICECCE)*, 1–6, DOI: 10.1109/ICECCE63537.2024.10823560 (2024).

51. Chuan Wang, W., Can Tian, W., Mei Xu, D. & Fei Zang, H. Arctic puffin optimization: A bio-inspired meta-heuristic algorithm for solving engineering design optimization. *Advanced Engineering Software* **195**, 103694, DOI: https://doi.org/10.1016/j.advengsoft.2024.103694 (2024).

52. Huettmann, F., Artukhin, Y., Gilg, O. & Humphries, G. Predictions of 27 arctic pelagic seabird distributions using public environmental variables, assessed with colony data: a first digital IPY and GBIF open access synthesis platform. *Marine Biodiversity* **141**, 141–79, DOI: https://doi.org/10.1007/s12526-011-0083-2 (2011).

53. Burnham, K. K., Burnham, J. L., Johnson, J. A. & Huffman, igratory movements of Atlantic puffins Fratercula arctica naumanni from high Arctic Greenland *Plos one* **16**, e0252055, DOI: https://doi.org/10.1371/journal.pone.0252055 (2021).

54. Wang, Y., Ran, S. & Wang, G.-G. Role-oriented binary grey wolf optimizer using foraging-following and lévy flight for feature selection. *Applied Mathematical Modelling* **126**, 310–326, DOI: https://doi.org/10.1016/j.apm.2023.08.043 (2024).

55. Mirjalili, S. & Lewis, A. S-shaped versus v-shaped transfer functions for binary particle swarm optimization. *Swarm and Evolutionary Computation* **9**, 1–14, DOI: https://doi.org/10.1016/j.swevo.2012.09.002 (2013).

56. Ahmadianfar, I., Bozorg-Haddad, O. & Chu, X. Gradient-based optimizer: A new metaheuristic optimization algorithm. *Information Sciences* **540**, 131–159, DOI: https://doi.org/10.1016/j.ins.2020.06.037 (2020).

57. Arora, S. & Singh, S. Butterfly optimization algorithm: a novel approach for global optimization. *Soft Computing* **23**, 715–734, DOI: https://doi.org/10.1007/s00500-018-3102-4 (2019).

58. Mirjalili, S. Sca: a sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems* **96**, 120–133, DOI: https://doi.org/10.1016/j.knosys.2015.12.022 (2016).

59. Abdel-Basset, M., Mohamed, R., Azeem, S. A. A., Jameel, M. & Abouhawwash, M. Kepler optimization algorithm: A new metaheuristic algorithm inspired by kepler's laws of planetary motion. *Knowledge-Based Systems* **268**, 110454, DOI: https://doi.org/10.1016/j.knosys.2023.110454 (2023).

60. Rajabioun, R. Cuckoo optimization algorithm. *Applied Soft Computing* **11**, 5508–5518, DOI: https://doi.org/10.1016/j.asoc.2011.05.008 (2011).

61. Mirjalili, S. *et al.* Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. *Advanced Engineering Software* **114**, 163–191, DOI: https://doi.org/10.1016/j.advengsoft.2017.07.002 (2017).