Behavior Analysis of the (DESFO) Algorithm Using Different Transfer Functions and Different Classifiers

Ahmed Sabry Abolaban [1*], O. E. Emam [1] and Safaa. M. Azzam [1]

[1] *Department of Information Systems, Faculty of Computers and Artificial Intelligence, Helwan University, P.O. Box 11795, Helwan, Egypt.*

Correspondence

[*]*Ahmed Sabry Ahmed Abdallah, Department of Information Systems, Faculty of Computers and Artificial Intelligence, Helwan University, P.O. Box 11795, Helwan, Egypt.*

Email: Ahmed_sabry_1053@Fci.Helwan.edu.eg

Abstract

The process of feature selection (FS) plays a crucial role in optimizing model performance in machine learning. It achieves this by reducing data dimensionality and eliminating irrelevant features. This study evaluates the effectiveness of the Differential Evolution Sailfish Optimizer (DESFO) in FS using Transfer functions and Machine learning classifiers. The DESFO algorithm is tested with both V-shaped and S-shaped transfer functions across various classifiers, such as k-nearest Neighbors (k-NN), Support Vector Machine (SVM), and Random Forest (RF). The study conducted experiments on 14 benchmark datasets from the UCI Machine Learning repository, illustrating that using DESFO with either type of transfer function notably improves classification accuracy and computational efficiency. The findings indicate that V-shaped transfer functions perform well in situations requiring precise feature selection, whereas S-shaped transfer functions show outstanding generalization capabilities. When inspecting the mean accuracy, DESFO-RF with V-shaped V4 and DESFO-SVM with S-shaped V4 configurations outperform all other transfer functions in 10 of the 14 benchmarks. Regarding the mean Fitness functions, DESFO-RF with V-shaped V1 and DESFO-SVM with S-shaped V3 is superior in 8 of the 14 benchmarks. Moreover, considering the number of selected features, DESFO-RF equipped with an S-shaped V1 stands out in 8 of 14 benchmark datasets.

Keywords:

Feature selection, Transfer Functions, V-shaped, S-shaped, Classification, Machine learning, differential evolution, Sailfish, optimization

1. Introduction:

Machine learning (ML) algorithms use features, or measurable aspects of the observed process, for classification. The feature count in ML applications has grown from tens to hundreds over time. Addressing the challenge of irrelevant and redundant variables, Feature Selection (FS) is a technique that helps in understanding data, reducing computational needs, overcoming the curse

of dimensionality, and improving prediction accuracy. Supervised learning algorithms utilize data as a matrix [1]. Optimizing high-dimensional datasets often requires reducing these datasets. This can be achieved by finding a similar matrix with fewer features, making it easier to use. Dimensionality reduction simplifies the process by focusing on discoveries with fewer columns. Additionally, selecting appropriate features can lower measurement costs and enhance problem understanding [2]and [3].

Due to its numerous benefits, FS is widely used in real-world applications, particularly in classification and regression problems. It has effectively addressed challenges in various domains, such as microarray analysis, image classification, facial recognition, and text classification [4].

Feature selection in (ML) involves using various statistical methods such as filters, wrappers, and embedded methods. The filter method is a preprocessing step that selects features based on specific criteria without considering their effect on the algorithm's performance [5] and [6]. On the other hand, wrapper methods evaluate feature subsets based on the accuracy of a given predictor and use search strategies to yield nested subsets of variables [7]. Finally, embedded methods perform variable selection during the training process and are specific to certain learning machines, making it impossible to separate the learning and (FS) steps [8]. Methods for (FS), such as wrappers or embedded methods, involve utilizing nonparametric algorithms like decision trees, support vector machines, and neural networks [9].

Over the past few decades, numerous techniques have been introduced for classification. Some of the most commonly utilized methods include K-Nearest-Neighbors (KNN), artificial neural networks (ANNs), support vector machines (SVMs), and ensembles of classification trees like random forest (RF) [10].

ML algorithms are found to be more precise when compared to statistical techniques such as logistic regression or discriminant analysis, particularly when the input datasets have varying statistical distributions or when the feature space is complicated [11] and [12]. In recent times, with the increase in computational power, MLAs have gained more attention, leading to an improvement in the quality of pattern recognition systems. Hence, most classification studies report that RF, KNN, and SVM are the top classifiers, as they achieve high accuracies [13].

Numerous advanced literature reviews have emphasized the difficulties faced in (FS) across diverse (ML) domains. As per Zhigljavsky [14], FS is likely a combinatorial optimization problem that falls under the NP-complete category because of the exponential increase in potential solutions with adding more dataset features. This makes it challenging to find feature subsets that are close to optimal. FS has been classified as an NP-hard problem that exhibits an exponential increase in computational time with complexity [15] and [16]. As a result, considerable attention has been given to using metaheuristic (MH) algorithms, which effectively optimize diverse scenarios [17].

There are four main types of algorithms used in MH: SI algorithms, EA, PhA, and Human-based algorithms [18] and [19]. SI algorithms are based on animal behaviors and swarms, such as particle swarm optimization (PSO) and ant colony optimization (ACO), and are commonly used in vehicle routing and FS applications. The Artificial Bee Colony (ABC) algorithm is another type of SI algorithm [20]. Evolutionary algorithms (EA) are designed to replicate the natural process of evolution, including mutation and selection. Examples of such algorithms include Genetic Algorithm (GA) and Differential Evolution (DE. Physical laws inspire some algorithms in the field of PhA. These algorithms include BBBC, GSA, and MVO. Algorithms based on human behaviors, like TBLA, SELOA, sine cosine algorithm, and volleyball Premier League Algorithm, are designed

to optimize various processes [21]. MH algorithms follow a similar pattern that involves two key phases: exploration and exploitation. These algorithms randomly generate operators to explore the problem space during the exploration phase. In the exploitation phase, they attempt to find the best solution by balancing these phases. This approach helps prevent getting stuck in local optima traps.

When dealing with the (FS) tasks, it becomes crucial to incorporate Transfer functions (TFs). According to various studies, TFs are highly recommended for several reasons. To begin with, TFs are not bound to any particular algorithm and don't influence an algorithm's search behavior. Furthermore, the algorithm's computational complexity remains unaffected since the TF is computed for each solution in every iteration. Finally, using a TF can enhance both exploration and exploitation [22], [23] and [24].

1.1. motivation

The (DE) algorithm was proposed by Storn et al.[25] As a stochastic search approach that operates on populations. It is a practical and straightforward approach for globally optimizing continuous search, and it has also been utilized in diverse fields. The Sailfish Optimizer (SFO) was created and introduced by Shadravan et al. [26]. It depends on the behavior of a flock of sailfish hunting a flock of sardines. It imitates their hunting strategy by attacking the flock of sardines and retrograding after capturing their prey. This algorithm operates on the principle of population and aims to optimize performance. It has become well-liked in optimization because of its efficiency and robustness. Consequently, this paper introduces a hybrid technique named DESFO algorithm with eight (TFs) and three (ML) classifiers, which combines both algorithms for (FS).

To apply the DESFO algorithm to solve an FS problem, a mapping function must change continuous values obtained by the DESFO algorithm into binary values of either 0 or 1. These binary values represent the decision variables [27]. (TFs) [24] determine how quickly the decision variables (DV) value range from 0 to 1 and vice versa.

In ML, k-nearest Neighbor (k-NN), Random Forest (RF), and Support Vector Machine (SVM) are commonly used (ML) classification algorithms. This study uses DESFO as a search optimization approach to identify the most relevant features. As fitness evaluators, it employs various classifiers, including k-NN, RF, and SVM. Using these evaluators, a new wrapper FS method is composed [27].

Meta-heuristics have been quite successful in (FS). However, many existing methods have focused solely on the k-NN classifier while neglecting the SVM in several instances. The Random Forest (RF) technique has received minimal focus, even though SVM and RF generally produce superior results compared to k-NN in a variety of classification tasks [28] and [29].

1.2. Contribution

This paper proposes applying the DESFO algorithm, Transfer functions (TFs), and three (ML) classifiers. It introduces innovative contributions, which are summarized as follows:

1. The DESFO algorithm is described by incorporating and replicating DE and SFO.
2. The method utilizes eight (TFs), specifically the V-shaped and S-shaped functions, to convert position values into binary values.
3. The DESFO algorithm is used for wrapper (FS) in supervised classification.
4. The performance of DESFO with (TFs) and classifiers is assessed using measures such as the mean of fitness, accuracy, and the mean count of features chosen.

5. The proposed DESFO algorithm's performance is evaluated using K-NN, RF, and SVM machine learning classifiers with the mentioned metrics.

1.3. Structure

Section 2 discusses the latest findings and critiques in art and literature. Section 3 outlines the prior research conducted. Section 4 describes the methods behind the newly proposed DESFO algorithm, including using the eight (TFs) and other related procedures. Section 5 showcases the outcomes of experiments, comparing the performance of the multi variants of the DESFO algorithm with (TFs) and (ML) classifiers. Finally, Section 6 summarizes the conclusions of the study.

2. Related Work

Mafarja et al.[22] introduced the Binary Dragonfly Algorithm (BDA). The BDA's essence is a Transfer Function (TFs) that transforms continuous search spaces into discrete ones. This study introduced S-shaped and V-shaped TFs to the BDA and evaluated them on eighteen UCI datasets. The key innovation for balancing exploration and exploitation is the creation of dynamic S-shaped and V-shaped TFs to fine-tune the step vector's impact.

Faris et al. [30] introduced a new binary version of the Salp Swarm Algorithm (BSSA) and two new SSA-based FS strategies. The first converts SSA to binary using eight TFs, and the second presents a crossover operator to improve exploration. Their effectiveness was tested on 22 UCI datasets against five FS techniques.

Too et al. [31] introduced a binary version of the HHO algorithm, named BHHO, to address the (FS) challenges in classification problems. The new BHHO algorithm incorporates either an S-shaped or V-shaped (TFs) for transforming continuous variables into binary form. Furthermore, they developed another variant, the quadratic binary Harris hawk optimization (QBHHO), to improve BHHO's efficacy. To evaluate the performance of these proposed algorithms, the study utilized twenty-two datasets from the UCI (ML) repository.

Mafarja et al. [32] introduced a model named HBALO; this methodology merges QuickReduct and CEBARKCC with the binary ant lion optimizer, refining initial random solutions with two filter techniques. After enhancement, the BALO algorithm selects the optimal solution. Tested on 18 UCI datasets, this binary approach was benchmarked against recent methods.

Chen et al. [33] introduced an enhanced version of the dragonfly algorithm. The approach improves the dragonfly algorithm by incorporating a BDA-DDO that adjusts step size adaptively and introduces a novel differential operator for quicker convergence. It includes a directed variant for targeted searches and an adaptive method to diversify populations, tested successfully on 14 UCI datasets.

Chantar et al. [34] introduced an advanced binary grey wolf optimizer (GWO) incorporated into a wrapper FS strategy for addressing issues in classifying Arabic texts. This modified binary GWO is employed as a wrapper-based (FS) method. The effectiveness of this method was assessed across various learning models, such as decision trees, K-nearest neighbor, Naive Bayes, and SVM classifiers. Three public Arabic datasets were used for evaluation to determine the performance of different BGWO-based wrapper approaches.

Prudhvi et al. [35] introduced an approach combining Binary PSOGSA with a Radial Basis Neural Network, utilizing a unique fitness function for effective FS. This ensures minimal features

while balancing sensitivity and specificity. It proved efficient in selecting optimal features when applied to healthcare and finance datasets.

Hussien and colleagues [36] developed a binary version of the Whale Optimization Algorithm (BWOA) for efficient FS and classification. The algorithm converts whale positions into binary with an S-shape sigmoid function. The method simplifies dimensions and employs a K-NN classifier for feature validation.

Fatahi et al. [37] introduced an improved version of the Binary Quantum-based Avian Navigation Optimizer Algorithm; IBQANA updates the Binary Quantum-based Avian Navigation Optimizer for better Feature Subset Selection in medical data, fixing past inefficiencies. It introduces the Hybrid Binary Operator (HBO) for accurate binary transitions of continuous values and the Distance-based Binary Search Strategy (DBSS) for improved search efficiency and faster convergence by blending exploration and exploitation with a variable probability approach to dodge local optima.

3. Preliminary work

3.1 DESFO Algorithm

The No Free Lunch Theorem (NFL) [38] indicates that an algorithm capable of optimally solving all optimization issues does not exist. An algorithm's capability for FS varies with the dataset, indicating a need for better metaheuristic approaches to tackle FS challenges efficiently. The DESFO algorithm introduced by Azzam et al. [39], which combines Differential Evolution (DE) and Swarm Fish Optimization (SFO), aims to improve FS and classification accuracy, addressing an unmet need in current research.

3.1.1 DE

Storn et al. [25] Presented the (DE) algorithm, which is recognized for its effectiveness among Evolutionary Algorithms and is notable for quick convergence and simplicity. Utilizing just three parameters: Population size (NP), Crossover rate (Cr), and Scaling Factor (F), DE effectively solves a wide array of optimization problems. It starts with an initial solution set, generating new solutions by modifying existing ones based on the weighted differences between pairs of other solutions, also known as mutant solutions. The (DE) algorithm has proven effective and has been embraced for addressing optimization problems in diverse domains [40].

The primary operators and the overall architecture of DE algorithm are listed as follows:

3.1.1.1Mutation:

In every iteration (t), Differential Evolution (DE) uses a mutation operator to create a new donor vector or (mutant) vector for each solution. This operator chose three candidate solutions randomly, Emphasizing that a mutant or donor vector is created by multiplying the difference between two vectors by a scale factor as shown in Eq. (1), followed by adding this scaled difference to a third solution [25].

$$V_{i,G+1} = x_{r1,G} + F\left(x_{r2,G} - x_{r3,G}\right) \quad (1)$$

Where three unique integers $r1, r2, and\ r3$ are chosen at random, each lying within the range of 1 to NP, and NP is integer and positive, which is four or more. Moreover, these integers' values are distinct from the current index, denoted as i.

After that, the differential amplification $(x_{r2,G} - x_{r3,G})$ is boosted by a constant factor F, which can vary between [0, 2].

3.1.1.2 Crossover:

After mutation, a new (trial) vector offspring is created from the solution using a crossover search operator. The exponential and binomial operators are commonly employed as direct crossover search methods. It's important to note that this applies to every decision variable denoted by (DV) j. Where ($rand \leq C_r$), as the Eq. (2):

$$u_{i,j,G} = \begin{cases} u_{i,j,G} & if\ rand\ (j) \leq C_r\ or\ j = j_{rand} \\ x_{i,j,G} & otherwise'\ \ j = 1,2, \dots D \end{cases} \qquad (2)$$

A value of $rand\ (j)$, denoted as the "jth evaluation" and chosen at random, is chosen randomly in the range of [0, 1]. This process guarantees to acquire a minimum of one trial vector for the design variable (DV). C_r Indicates crossover rate is crucial for determining the variable count, is sourced from the (donor vector), and it is ensured that $V_{i,G+1}$provides at least one parameter to $u_{i,j,G}$

3.1.1.3Selection:

An operator for selection is used to identify the best solution by comparing the objective function values of both the parent and the offspring. If the offspring exhibits a decreased objective function value, it is retained for future iterations. Conversely, if this is not the case, the parent vector remains in the current generation and is generated by adapting Eq. (3).

$$x_{i,G+1} = \begin{cases} u_{i,G} & if\ \left(f(u_{i,G}) \leq (x_{i,G})\right) \\ x_{i,G} & otherwise \end{cases} \qquad (3)$$

To decide whether to include it in the new generation (G + 1), $x_{i,G+1}$which is the trial vector is compared to the target vector $x_{i,G}$ by applying the greedy criterion. Should the (trial vector) $x_{i,G+1}$yield a cost function value lower than that of the target vector $x_{i,G}$, then the trial vector $x_{i,G+1}$ takes the place of the target vector $x_{i,G}$. Otherwise, the value of the original target vector is maintained.

3.1.2 SFO

Shadravan et al [26]. Introduced an innovative algorithm known as the (SFO), in which the injured sardine that showcases the optimal fitness value is marked, pinpointing its location by (P^i_{srdinj}) during i^{th} iteration. With every iteration, both the sardines's and sailfish's locations are adjusted. For the iteration, it revises the position of a sailfish by utilizing information from the sailfish named (elite fish) $PSlfbesti$ and the (injured sardine's fish) according to a predetermined criterion.

In each iteration, the locations of sailfish and sardines are updated, and the new position is represented by $i+$. The (elite) and the (injured) are responsible for adjusting a sailfish's position to a new one, represented by P^{i+1}_{Slf} as shown in Eq. (4)

$$P_{Slf}^{i+1} = P_{Slfbest}^{i} - \sigma_i \left(rnd * \frac{P_{Slfbest}^{i} + P_{srdinj}^{i}}{2} - P_{Slf}^{i} \right) \quad (4)$$

Where $rnd \in (0,1)$ is determined randomly, and σ_i is a coefficient derived by Eq. (4):

$$\sigma_i = (3 * rand * PrD - PrD) \quad (5)$$

In every iteration, the density of prey PrD , indicative of the available prey count, is calculated using Eq. (6). As the prey count diminishes with group hunting activities, the value accordingly decreases.

$$PrD = 1 - \frac{N_{Slf}}{N_{Slf} - N_{srd}} \quad (6)$$

The numbers for sailfish and sardines are denoted by N_{Slf} and $N_{srd,}$ respectively, and can be determined using Eq. (7):

$$N_{Slf} = N_{srd} * Prcent \quad (7)$$

$Prcent$ Represents the population of sardine that comprises the original population of sailfish.The initial amount of sardines is assumed to be higher than the initial amount of sailfish
In each iteration, the locations of the sardines are adjusted following Eq. (8):

$$P_{Srd}^{i+1} = rand * (P_{Slfbest}^{i} - P_{Srd}^{i} + ATK) \quad (8)$$

The old and new locations of the sardine are denoted by P_{Srd}^{i} and P_{Srd}^{i+1}, respectively. Meanwhile, the representation of the strength of the sailfish's attack ATK at one-by-one iteration is determined by Eq. (9):

$$ATK = A * \left(1 - (2 * itr * k)\right) \quad (9)$$

ATK plays a pivotal role in dictating how many sardines adjust their positions and how far they move. Reducing ATK can facilitate the convergence of search agents. The determination of the quantity of sardines adjusting their positions and variables number related to the sardines are calculated using Eq. (10) and (11):

$$\gamma = ATK * N_{Srd} \quad (10)$$

$$\delta = ATK * v \quad (11)$$

Where (N_{Srd}) denotes the sardines and (v) refers to the variables, whenever a sardine exceeds a sailfish fitness level, the sailfish will change its location to pursue the sardine. However, this results in the sardine being eliminated from its population.

3.2 Transfer (Mapping) Functions (TFs)

The DESFO algorithm generates a solution consisting of continuous values; it can't be directly applied to a (FS) problem without modifications. Thus, a mapping or (transfer) functions are required for converting continuous values to binary format, 0 or 1. (TFs)[24] Dictate how decision variables transition between 0 and 1. When choosing a TF for converting continuous values to binary, several key factors need to be considered:

- Values derived from the (TF) should fall between 0 and 1, indicating the agent's likelihood of altering its present position.
- Should the alarm value warning (alarm) value fall below the safety threshold ST, the likelihood of the TF changing its current position in the subsequent iteration should increase. This is under the assumption that an agent with a warning (alarm) value exceeding ST is likely veering too far from the best solution.
- In case the warning (alarm) value is low, the TF should offer a low likelihood of altering the current position.
- As the alarm value approaches ST, the TF-induced likelihood should increase, encouraging agents to correct their course and quickly return to their previous best position in future iterations.

  The concepts demonstrate the significant ability of TFs to convert the ongoing search process into binary form for each Y, as described by Eq. (12):

$$\left(y_{i,j}^{t+1}\right)_{bin} = \begin{cases} \begin{cases} -\left(y_{i,j}^{t}\right)_{bin} \; if \; rnd < TF(y_{i,j}^{t+1}) \\ \left(y_{i,j}^{t}\right)_{bin} \; if \; rnd > TF(y_{i,j}^{t+1}) \end{cases} & if \; TF \; is \; V-shape \\ \begin{cases} 0 \; if \;\; rnd < TF(y_{i,j}^{t+1}) \\ 1 \; if \; rnd > TF(y_{i,j}^{t+1}) \end{cases} & if \; TF \; is \; S-shape \end{cases} \tag{12}$$

Where, $\left(y_{i,j}^{t+1}\right)_{bin}$ represents the value in the j-th dimension for individuals in the current iteration$t+1$, and rnd is a randomly chosen number [0, 1], and $TF(y_{i,j}^{t+1})$ represents the probability value determined by employing a specific (TF) to each continuous value of the *j-th* component of agent i. According to Equation (6), we encounter two scenarios: (first) if the TF has an S-shape, in case of rnd is less than the value of the given probability, the *j-th* dimension of the individual is updated to 0; if not, it is updated to 1; also in case of the TF is V-shaped, then if rnd is less than the provided value of probability, the value of the *j-th* dimension is inverted; if not, it remains the same. Therefore, applying S-shaped and V-shaped TFs, as detailed in Table 1, can efficiently convert continuous variables to binary values.

Table 1 presents two variants or families of (TFs), whereas Figure 1 displays these variants, categorizing them into S-shaped and V-shaped (TFs).

Table 1:V-shaped and S-shaped TFs [24]

V-shaped family		V-shaped family	
V-TF version	TF	S-TF version	TF
V1	$TF_{(y)} = \left\|\text{erf}(\frac{\sqrt{\pi}}{2})y\right\| = \left\|\frac{\sqrt{2}}{\pi}\int_0^{(\frac{\sqrt{2}}{\pi})y} e^{-e^{-t^2}}dt\right\|$	S1	$TF_{(y)} = \frac{1}{1+e^{-2y}}$
V2	$TF_{(y)} = \|\tanh(y)\|$	S2	$TF_{(y)} = \frac{1}{1+e^{-y}}$
V3	$TF_{(y)} = \left\|y/\sqrt{1+y^2}\right\|$	S3	$TF_{(y)} = \frac{1}{1+e^{(-\frac{y}{2})}}$
V4	$TF_{(y)} = \left\|\frac{2}{\pi} arc\tan(\frac{\pi}{2}y)\right\|$	S4	$TF_{(y)} = \frac{1}{1+e^{(-\frac{y}{3})}}$

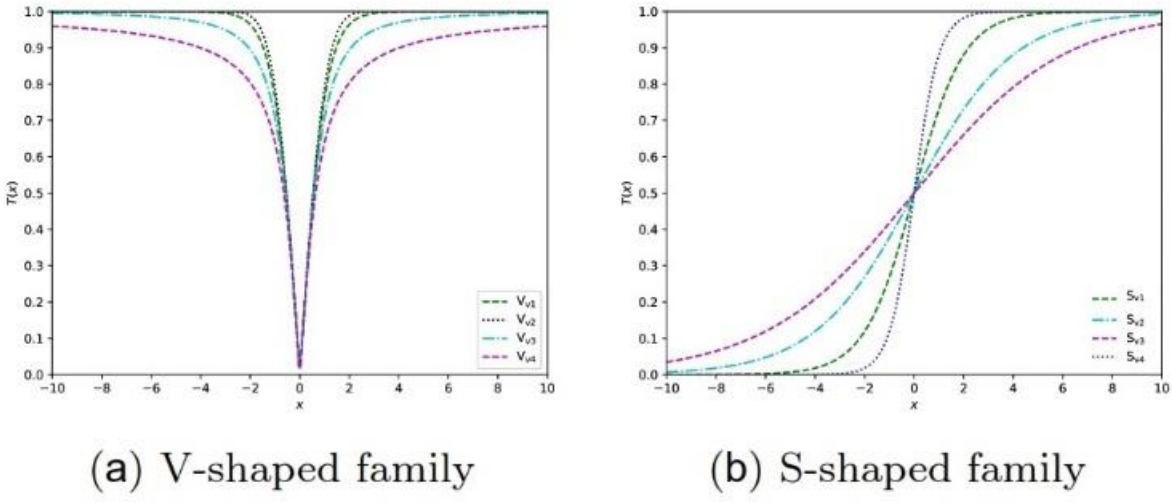

Figure 1: V-shaped and S-Shaped TF families

3.3 Machine Learning (ML) Classifiers:

In this study, we utilized three of the most effective ML classifiers for FS purposes: K-NN, SVM, and RF. Each of these methods is detailed in the subsequent subsections provided.

3.3.1 K-Nearest Neighbor classifier (k-NN):

The k-nearest Neighbor (k-NN) classifier [41] is widely recognized for its power in pattern recognition and ML fields. Its ease of implementation makes it a preferred choice over more complex supervised learning techniques [42]. K-NN is widely used in fields like healthcare, forestry, image/video analysis, and finance for its ability to classify patterns. It works by creating classification rules from training data, and it assigns labels to unlabeled data in test sets based on the nearest training samples. Choosing the right "k" is crucial for its accuracy and is typically found through trial and error. In the empirical studies conducted, the k-NN classifier, utilizing a Euclidean distance metric, is tested with a k value of 5 [43] and [22], and the chosen feature subsets are evaluated for their effectiveness.

3.3.2 Random Forest classifier (RF):

The Random Forest (RF) [44] algorithm is an ML classifier widely used in various computing tasks such as label distribution learning, action recognition, detection, visual tracking, facial expression analysis, image classification, and time series forecasting. It consists of multiple decision trees and is known for its robustness against data labeling errors, ability to handle multiple classes, FS support, parallel computation, minimalism in tuning parameters, and efficient handling of numerical and categorical data. RF remains popular for its simplicity, interpretability, computational efficiency, and its method of improving classification by breaking down data into smaller subsets for optimized learning[45], [46], and [47].

RF algorithm performance, particularly for real-time applications, is affected by the number of trees and their maximum depth. A hyper-heuristic approach can optimize these parameters, enhancing speed and accuracy. In tests, using ten estimators with a maximum depth of 5 showed significant improvements in classification accuracy. Despite some drawbacks, the RF model's capability to accurately model complex relationships demonstrates its value.

3.3.3 Support Vector Machine classifiers (SVM):

The Support Vector Machine (SVM) algorithm [48] stands as a prominent wrapper-based classifier utilized in data science, known for effectively segregating multiple classes. The technique relies on using hyper-planes to separate different groups. A significant benefit of SVM is its ability to provide reliable accuracy with low computational effort. It achieves this by applying a non-linear function, ϕ, to shift the original data into a higher dimensional space, where it seeks to linearly divide the data along a hyper-plane that maximizes separation margins between classes. However, challenges include choosing the proper base function and fine-tuning its parameters [49]. Finding the optimal decision plane essentially becomes an optimization problem, where a kernel function is tasked with determining the most suitable higher-dimensional space for achieving linear division of categories via a non-linear transformation.

4. Methodology of the proposed DESFO

This study introduces a method that combines eight (TFs), including V-shaped and S-shaped families, with the DESFO, merging the DE and SFO algorithms. This approach for FS uses K-NN, RF, and SVM in several steps: initialization, position updates, binary mapping or conversion, and fitness assessment. The DESFO algorithm runs for 100 iterations, split between DE and SFO at 50 iterations for both. Initially, DE takes the lead for the first 50 iterations, focusing on optimizing to find the best solution, which is then further refined by SFO to improve classification accuracy by selecting relevant features. Each phase is described in detail in subsequent sections.

4.1 Initial Population Generation

The initial stage of applying the DESFO algorithm involves creating a starting set of Y positions, symbolizing possible solutions within a space defined by D dimensions. The size of this population is established based on a particular formula as shown in Eq. (13):

$$Y = Round\left(10 + 2 * \sqrt{D}\right). \quad (13)$$

Y represents the overall count of positions, whereas D symbolizes the dimensionality of the issue. The matrix describing positions is defined as follows:

$$k = \begin{bmatrix} k_{1,1}, k_{1,2}, \dots\ k_{1,p} \\ k_{2,1}, k_{2,2}, \dots\ k_{2,p} \\ \vdots \quad \vdots \quad \ddots \quad \vdots \\ k_{Y,1}, k_{Y,2}, \dots\ k_{Y,p} \end{bmatrix}$$

The solution j^{th} is denoted by, where j stands for the component. Initially, the population, symbolized by K, is created within specific boundaries as follows in Eq. (14):

$$K_i^u = u(0,1) * (UB - LB) + LB \qquad (14)$$

4.2 Position Update in DESFO algorithm

One must employ the formulas for DE and SFO algorithms for position updating, detailed in sections 3.1.1 and 3.1.2. Following the update, the position is subject to binary values, as outlined in section 4.3.

4.3 Binary converting

Converting continuous positional values to binary is important to using the FS method effectively. This is because the DESFO method calculates positional values and works differently than FS's binary structure. FS uses a binary vector to mark selected features with 1s and unselected ones with 0s. The number of elements in this vector equals the number of features in the initial dataset.

(TFs) in Table 1 have been employed with the DESFO. The value of the position obtained is symbolized by P , with a position in DESFO deemed to have a legitimate TF output if it is below 0.64 and lies within the interval [0, 1]. The prescribed method for updating the position in DESFO follows the Eq. (15):

$$P_i^{bin} = \begin{cases} 1, & If\ rnd < v(P_i) \\ 0, & otherwise \end{cases} \qquad (15)$$

4.4 Fitness Evaluation

Balancing feature set size and accuracy is crucial. A smaller feature set requires more precise classifiers like k-NN, RF, and SVM but risks reducing accuracy due to fewer features[49]. This implies a trade-off between the feature set size and the selection of optimal features, indicating a balance might be necessary between accuracy and feature set magnitude.

When evaluating an algorithm's performance, it's important to consider the balance between accuracy and feature size, as expressed in Eq. (16):

$$FIT = \alpha_1 * (1 - accuracy) + \alpha_2 + \left|\frac{|D^*|}{|D|}\right| \qquad (16)$$

The two coefficients, α1 and α2 representing the weight coefficients, are present in the specified equation. *α1* varies in the range from 0 to 1. *α2* is obtained by subtracting α1 from 1. The coefficients were established after comprehensive experimentation, and they expressed the

proportion of chosen features relative to the total feature count in the initial dataset. The symbol |D*| represents the number of the chosen features, whereas |D| signifies the entire features present within the initial dataset.

4.5 The flowchart of DESFO :

Figure 2 shows, the steps and procedures of the DESFO algorithm.

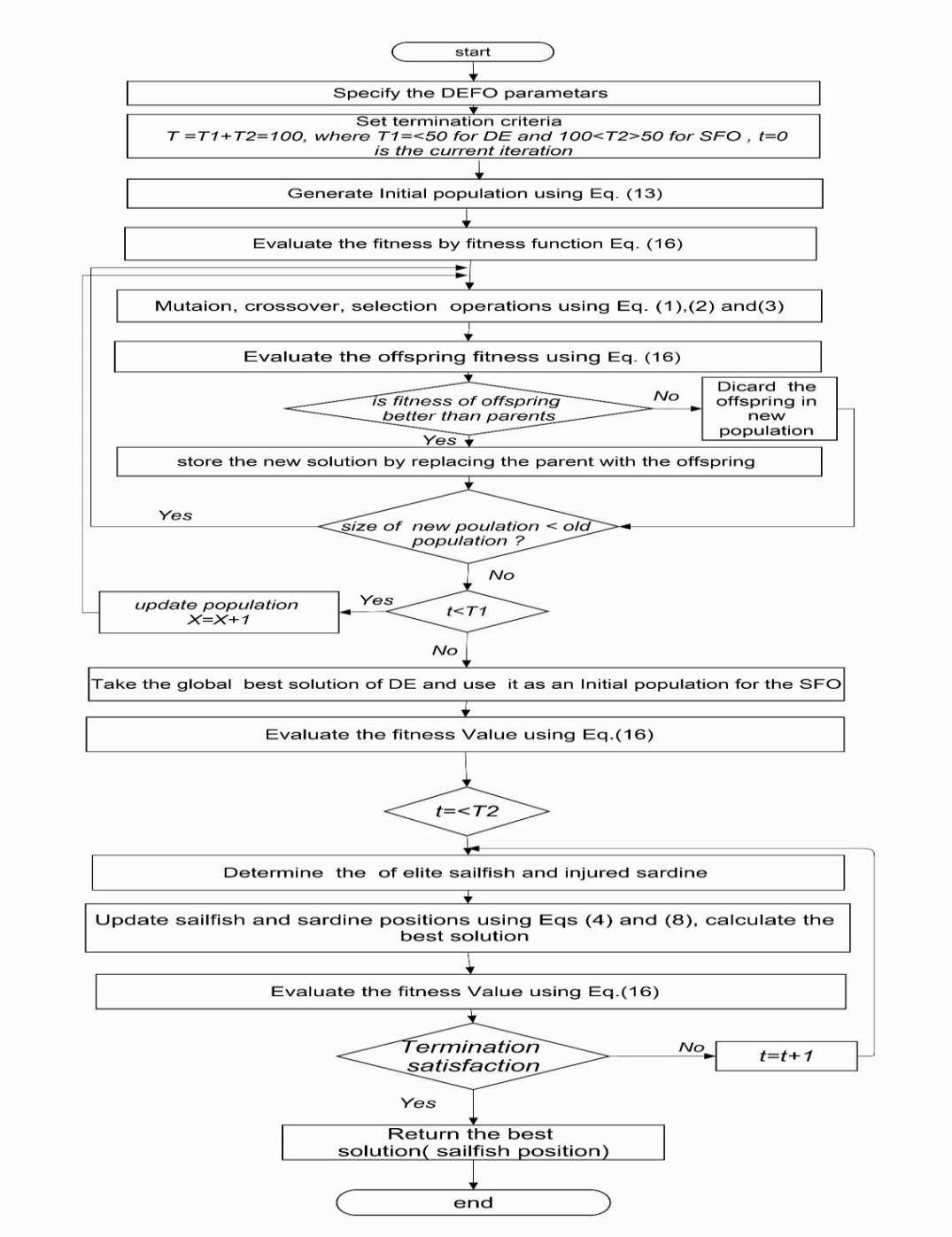

Figure 2: DESFO flowchart

4.6 Analysis of DESFO complexity:

To analyze the DESFO algorithm's complexity, delving into the computational processes involved is needed.

Breakdown of Complexity to the following:

1. Initialization:
- Establishes *NP* individuals, each showcasing *D*.
- Complexity: *O* x *(NP×D).*

2. Steps in the DE Algorithm:
a. The Mutation phase:

- In the mutation phase, each individual selects three distinct individuals and calculates the difference between their vectors.
- Complexity per individual: Total complexity for this mutation phase: *O × (NP×D).*

b. The Crossover Phase:

- Each undergoes a crossover based on a CR probability.
- Complexity: *O × (NP×D).*

c. The Selection phase:

- Calculating their fitness level is necessary to decide between the target and trial vector.
- Complexity: *O × (NP).*

Updates on the SFO Algorithm:
1. Position Adjustment:
- Every Sailfish adjusts its position considering the locations of the leading and the weaker sardines.
- Generation complexity: *O × (NP)*

2. Binary transferring and fitness assessment:
a. Binary transferring:
- The D characteristics of all the NP elements are converted from a numerical value into a binary representation using specific transformers (TFs).
Overall complexity: *O × (NP×D)*

b. Fitness assessment:
- The method to evaluate performance varies with the classification algorithm applied, whether it be k-NN, RF, or SVM. This might be influenced by the amount of attributes D,
The complexity: *O × (NP × f (D))*, where *f (D)* represents the computational complexity for the assessment of a single entity.

The overall complexity for all generations, $MaxGens$ is
$(MaxGens \times (3 \times NP \times D + NP \times f\ (D)))$

5. Results and analysis

The experiments and results analysis using the DESFO algorithm are demonstrated in this section.

5.1 datasets Overview

To thoroughly assess and confirm the effectiveness of the methods introduced in this paper, we utilized 14 diverse, multi-scale benchmark datasets from the UCI data repository [50] spanning various domains (such as biology, politics, electromagnetic, gaming, physics, chemistry, and artificial intelligence) in all our experiments. These datasets play a crucial role in effectively substantiating the techniques proposed in this study, considering the varying numbers of instances and features they contain. The specifics of these datasets are presented in Table 2.

Table 2: Overview of the data collections utilized in our research.

Benchmark	#. Records	#. Features	Domain
BreastCancer	699	9	Biology
BreastEW	569	30	Biology
Exactly2	1000	13	Biology
IonosphereEW	351	34	Electromagnetic
KrVsKpEW	3196	36	Game
Lymphography	148	18	Biology
M-of-n	1000	13	Biology
PenglungEW	73	325	Biology
SonarEW	208	60	Biology
Tic-tac-toe	958	9	Games
Vote	300	16	Politics
WaveformEW	5000	40	Physics
WineEW	178	13	Chemistry
Zoo	101	16	Artificial

5.2 Evaluation metrics

The evaluation of The DESFO algorithm's effectiveness is carried out through eight (TFs) (both V-shaped and S-shaped) and involves the use of three highly regarded ML classifiers: K-NN, RF, and SVM. These evaluations are performed individually across 30 trials for each benchmark. Specific metrics are used to assess the (FS) approach.

- **Mean accuracy:**

Conducting the procedure independently across 30 iterations can establish the precise rate of data categorization (Me).

$$Mean_{acc} = \frac{1}{30}\frac{1}{m}\sum_{k=1}^{30}\sum_{r=1}^{m} match(PL_r, AL_r) \qquad (17)$$

Where $Mean_{acc}$ symbolizes the mean accuracy, whereas m represents the total number of samples within the testing subset. The symbol PLr denotes the predicted class label for a given sample,

while *ALr* represents the reference class label. A specific function, named match (*PLr, ALr*), is used to compare these two labels. If *PLr* matches *ALr*, the match (*PLr, ALr*) function returns a value of 1; if they do not match, the value is 0.

- **Mean fitness**:

The ($Mean_{Fit}$) is utilized to assess the average outcomes of fitness obtained by implementing the suggested method across 30 separate trials. The optimal outcome is represented by the smallest value, which is determined by evaluating the fitness as follows:

$$Mean_{Fit} = \frac{1}{30}\sum_{i=1}^{30} f_*^k, \quad (18)$$

The symbol ($Mean_{Fit}$) represents the average fitness value, whereas f_*^i signifies the optimal fitness result achieved in each iteration in the 30 i-th iterations.

- **Mean selected number of features**:

The metric symbolized by ($Mean_{Feat}$) stands for the mean number of selected features, calculated by independently executing the method 30 times and is described as follows:

$$Mean_{Feat} = \frac{1}{30}\sum_{i=1}^{30} \frac{|d_*^i|}{|D|}, \quad (19)$$

In the given context, where $|d_*^k|$indicates the chosen attributes and the count of attributes in the optimal solution for each of the i-th iterations, with "|D|" representing the total of feature number utilized from the benchmarks.

Python is utilized to execute operations within a computer system setup that includes a CPU, specifically an Intel i7 processor, 16 GB of RAM, and an NVIDIA GTX 1050i GPU.

5.3 DESFO behavior assessment utilizing eight TF:

DESFO was initially developed for continuous optimization and required adaptation for discrete space searches. Its effectiveness was evaluated using eight different transformation functions on 14 benchmark datasets to address the FS problem.

The various TFs underwent evaluation using DESFO alongside k-NN, RF, and SVM classifiers, focusing on the mean accuracy (μ_{Acc}), mean fitness value (μ_{Fit}), and average number of selected features μ_{Feat}. The outcomes of these evaluations are detailed in Tables 2 through 10.

The methods were developed using eight transcription factors (TFs), which are categorized into two groups: four V-shaped TFs labeled as Vv1, Vv2, Vv3, and Vv4, and four S-shaped TFs named Sv1, Sv2, Sv3, and Sv4. Thus, in the subsequent discussions, the proposed techniques are referred to as “DESFO-TF,” with TF representing any of the eight TFs mentioned. The abbreviations W,T, and L found at the bottom of the tables denote the number of times each method wins, ties, or loses in comparison to the other competitors. Through the examination and comparison of the results mentioned, the most suitable DESFO variant for each classifier will be determined based on the TF that shows the best performance for that particular classifier.

5.3.1 Evaluation of DESFO-TFs using the k-NN classifier:

Table 3 presents the $\mu_{Acc}$of DESFO across eight TFs, utilizing the k-NN classifier. DESFO-Vv4 significantly outperformed in 8 of the 14 datasets, and DESFO-Vv3, DESFO-Sv1, DESFO-Sv2, and DESFO-Vv1 following closely behind, excelling in 7, 7, 7, and 6 of 14 datasets, respectively. As a result, when considering the mean accuracy, DESFO-Vv4 is ranked highest among the evaluated methods.

Table 4 presents the mean Fitness Value μ_{Fit} for DESFO across eight TFs, utilizing the k-NN classifier. It was noted that DESFO-Sv4 demonstrated notable performance in 7 out of the 14 datasets, and DESFO-Vv3, DESFO-Sv3, and DESFO-Vv1 followed closely, showing significant results in 6, 6, and 5 out of 14 datasets, respectively. Hence, when considering the mean fitness values, DESFO-Sv4 is the top-performing method among all the TFs methods.

Table 5 presents the mean features selected μ_{Feat} for DESFO across eight TFs, using the k-NN classifier for analysis. Both DESFO-Sv1 and DESFO-Sv4 demonstrated remarkable performance in 7 out of the 14 datasets reviewed. Following them closely were DESFO-Sv3 and DESFO-Vv4, each showing significant outcomes in 6 and 5 out of the 14 datasets, respectively. Therefore, based on the average features selected, DESFO-Sv1 and DESFO-Sv4 emerge as the leading methods among all the TFs considered.

Table 3: The mean Accuracy μ_{Acc} results Using the eight TFs and K-NN with DESFO

Benchmarks	Eval-Metric	V-v1	V-v2	V-v3	V-v4	S-v1	S-v2	S-v3	S-v4
BreastCancer	μ_{Acc}	**0.9857**	**0.9857**	**0.9857**	**0.9857**	**0.9857**	**0.9857**	**0.9857**	**0.9857**
BreastEW	μ_{Acc}	0.9649	0.9649	0.9649	0.9649	0.9649	0.9649	**0.9658**	0.9649
Exactly2	μ_{Acc}	**0.7970**	0.7935	0.7880	0.7935	0.7880	0.7945	0.7870	0.7865
IonosphereEW	μ_{Acc}	0.9310	0.9310	0.9324	0.9324	**0.9563**	0.9493	0.9493	0.9535
KrVsKpEW	μ_{Acc}	0.9803	0.9788	0.9803	**0.9822**	0.9789	0.9819	0.9803	0.9814
Lymphography	μ_{Acc}	0.8333	0.8367	**0.8433**	0.8400	0.8333	0.8367	0.8400	0.8333
M-of-n	μ_{Acc}	0.9995	**1.0000**	**1.0000**	**1.0000**	0.9995	**1.0000**	0.9995	**1.0000**
PenglungEW	μ_{Acc}	0.6533	0.6533	0.6600	0.6533	**0.7333**	0.7067	0.6867	0.7200
SonarEW	μ_{Acc}	0.9786	0.9929	0.9857	0.9857	0.9857	0.9857	**0.9952**	0.9857
Tic-tac-toe	μ_{Acc}	**0.8542**	**0.8542**	**0.8542**	**0.8542**	**0.8542**	**0.8542**	**0.8542**	**0.8542**
Vote	μ_{Acc}	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
WaveformEW	μ_{Acc}	0.8445	0.8448	0.8454	**0.8480**	0.8436	0.8449	0.8453	0.8443
WineEW	μ_{Acc}	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
Zoo	μ_{Acc}	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
score	W	1	0	1	**1**	2	0	2	0
	T	5	6	6	**7**	5	6	5	6
	L	8	8	7	**6**	7	8	7	8

Table 4:The mean Fitness valueμ_{Fit} results Using the eight TFs and K-NN with DESFO

Benchmarks	Eval-Metric	V-v1	V-v2	V-v3	V-v4	S-v1	S-v2	S-v3	S-v4
BreastCancer	μ_{Fit}	**0.0201**	**0.0201**	**0.0201**	**0.0201**	**0.0201**	0.0204	**0.0201**	**0.0201**
BreastEW	μ_{Fit}	0.0366	0.0369	0.0366	0.0368	**0.0355**	0.0359	0.0356	0.0356
Exactly2	μ_{Fit}	**0.2062**	0.2097	0.2154	0.2093	0.215	0.2084	0.2155	0.2161
IonosphereEW	μ_{Fit}	0.0710	0.0714	0.0698	0.0698	**0.0443**	0.0516	0.0519	0.0472
KrVsKpEW	μ_{Fit}	0.0254	0.027	0.0256	**0.0232**	0.0267	0.0239	0.0256	0.025
Lymphography	μ_{Fit}	0.1691	0.1659	**0.1593**	0.1631	0.1694	0.1662	0.1625	0.1691
M-of-n	μ_{Fit}	0.0053	0.0048	0.0049	**0.0047**	0.0054	0.005	0.0055	0.005
PenglungEW	μ_{Fit}	0.3468	0.3466	0.34	0.3465	**0.2646**	0.292	0.3127	0.2777
SonarEW	μ_{Fit}	0.0248	0.0109	0.0177	0.0181	0.0174	0.0173	**0.0082**	0.0172
Tic-tac-toe	μ_{Fit}	**0.1544**	**0.1544**	**0.1544**	**0.1544**	**0.1544**	**0.1544**	**0.1544**	**0.1544**
Vote	μ_{Fit}	0.0022	0.0023	0.0023	0.0026	**0.0019**	**0.0019**	**0.0019**	**0.0019**
WaveformEW	μ_{Fit}	0.1596	0.1592	0.1591	**0.1563**	0.1607	0.1594	0.159	0.16
WineEW	μ_{Fit}	0.0032	0.0032	**0.0031**	**0.0031**	**0.0031**	**0.0031**	**0.0031**	**0.0031**
Zoo	μ_{Fit}	0.0033	0.0033	0.0034	0.0033	**0.0032**	0.0033	**0.0032**	**0.0032**
score	W	1	0	1	3	**3**	0	1	0
	T	2	2	3	3	**4**	3	5	5
	L	11	12	10	8	**7**	11	8	9

Table 5: The mean selected Features μ_{Feat} results Using the eight TFs and K-NN with DESFO

Benchmarks	Eval-Metric	V-v1	V-v2	V-v3	V-v4	S-v1	S-v2	S-v3	S-v4
BreastCancer	μ_{Feat}	**6.000**	**6.000**	**6.000**	**6.000**	**6.000**	6.300	**6.000**	**6.000**
BreastEW	μ_{Feat}	5.700	6.500	5.500	6.300	**2.400**	3.600	5.100	2.500
Exactly2	μ_{Feat}	6.800	6.900	7.200	6.300	6.600	6.500	**6.000**	6.100
IonosphereEW	μ_{Feat}	9.000	10.40	9.700	9.700	**3.500**	4.600	5.900	4.000
KrVsKpEW	μ_{Feat}	21.40	21.50	22.00	**20.10**	20.80	21.40	21.90	23.80
Lymphography	μ_{Feat}	**7.300**	7.500	7.600	8.500	7.900	8.100	7.400	**7.300**
M-of-n	μ_{Feat}	6.200	6.300	6.400	**6.100**	6.400	6.500	6.500	6.500
PenglungEW	μ_{Feat}	116.2	110.0	112.0	108.2	20.00	52.30	80.60	**17.10**
SonarEW	μ_{Feat}	21.80	23.10	21.10	23.60	19.40	18.70	20.90	**18.50**
Tic-tac-toe	μ_{Feat}	**9.000**	**9.000**	**9.000**	**9.000**	**9.000**	**9.000**	**9.000**	**9.000**
Vote	μ_{Feat}	3.500	3.700	3.600	4.100	**3.000**	3.100	**3.000**	3.100
WaveformEW	μ_{Feat}	22.80	**22.10**	24.00	23.20	23.40	23.50	23.20	23.30
WineEW	μ_{Feat}	4.100	4.100	**4.000**	**4.000**	**4.000**	**4.000**	**4.000**	**4.000**
Zoo	μ_{Feat}	5.200	5.300	5.400	5.200	**5.100**	5.200	**5.100**	**5.100**
score	W	0	1	0	2	**2**	0	1	**2**
	T	3	2	3	3	**5**	2	5	**5**
	L	11	11	11	9	**7**	12	8	**7**

5.3.2 Evaluation DESFO-TFs using the RF classifier:

Table 6 presents μ_{Acc} of the DESFO across eight TFs, utilizing the RF classifier. It was found that DESFO-Vv4 significantly outperformed 10 of the 14 datasets, with DESFO-Vv3 and DESFO-Sv4 closely behind, excelling in 9 and 8 of the 14 datasets, respectively. As a result, when considering the mean accuracy, DESFO-Vv4 is ranked highest among the evaluated methods.

Table 7 presents the mean Fitness Value μ_{Fit} for DESFO across eight TFs, utilizing the RF classifier. It was noted that DESFO-Sv1 demonstrated notable performance in 8 out of the 14 datasets, and DESFO-Vv3, DESFO-Sv3, and DESFO-Sv4 followed closely, showing significant results in 6, 6, and 5 out of 14 datasets, respectively. Hence, when considering the mean fitness values, DESFO-Sv1 is the top-performing method among all the TFs methods.

Table 8 presents the mean features selected μ_{Feat} for DESFO across eight TFs, using the RF classifier for analysis. DESFO-Sv1 demonstrated remarkable performance in 8 out of the 14 datasets reviewed. Following them closely were DESFO-Sv3 and DESFO-Sv4, each showing significant outcomes in 7 out of the 14 datasets for each of them. Therefore, based on the average features selected, DESFO-Sv1 emerges as the leading method among all the TFs considered.

Table 6: The mean Accuracy μ_{Acc} results Using the eight TFs and RF with DESFO

Benchmarks	Eval-Metric	V-v1	V-v2	V-v3	V-v4	S-v1	S-v2	S-v3	S-v4
BreastCancer	μ_{Acc}	**0.9857**	**0.9857**	**0.9857**	**0.9857**	**0.9857**	**0.9857**	**0.9857**	**0.9857**
BreastEW	μ_{Acc}	**0.9974**	0.9956	0.9930	0.9947	0.9939	0.9939	0.9956	0.9930
Exactly2	μ_{Acc}	**0.7650**	**0.7650**	**0.7650**	**0.7650**	**0.7650**	0.7645	**0.7650**	**0.7650**
IonosphereEW	μ_{Acc}	0.9704	**0.9732**	0.9704	**0.9732**	0.9718	0.9704	0.9704	0.9676
KrVsKpEW	μ_{Acc}	0.9486	0.9469	**0.9494**	0.9487	0.9467	0.9483	0.9470	0.9484
Lymphography	μ_{Acc}	0.8900	0.8833	**0.8933**	0.8933	0.8733	0.8833	0.8900	0.8767
M-of-n	μ_{Acc}	0.9935	0.9935	**0.9950**	0.9950	0.9825	0.9925	0.9870	0.9840
PenglungEW	μ_{Acc}	0.7533	0.7600	0.7467	0.7667	0.7667	0.7533	0.7600	**0.7933**
SonarEW	μ_{Acc}	0.9167	0.9238	0.9190	0.9286	0.9310	0.9286	0.9238	**0.9333**
Tic-tac-toe	μ_{Acc}	**0.8698**	**0.8698**	**0.8698**	**0.8698**	**0.8698**	**0.8698**	**0.8698**	**0.8698**
Vote	μ_{Acc}	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
WaveformEW	μ_{Acc}	0.8174	0.8176	0.8181	**0.8197**	0.8163	0.8196	0.8193	0.8193
WineEW	μ_{Acc}	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
Zoo	μ_{Acc}	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
score	W	0	0	1	**0**	0	0	0	0
	T	7	7	8	**10**	6	5	6	8
	L	7	7	5	**4**	8	9	8	6

Table 7: The mean Fitness value μ_{Fit} results using the eight TFs and RF with DESFO

Benchmarks	Eval-Metric	V-v1	V-v2	V-v3	V-v4	S-v1	S-v2	S-v3	S-v4
BreastCancer	μ_{Fit}	**0.0201**	**0.0201**	**0.0201**	**0.0201**	**0.0201**	0.0204	**0.0201**	**0.0201**
BreastEW	μ_{Fit}	0.0366	0.0369	0.0366	0.0368	**0.0355**	0.0359	0.0356	0.0356
Exactly2	μ_{Fit}	**0.2062**	0.2097	0.2154	0.2093	0.2150	0.2084	0.2155	0.2161
IonosphereEW	μ_{Fit}	0.0710	0.0714	0.0698	0.0698	**0.0443**	0.0516	0.0519	0.0472
KrVsKpEW	μ_{Fit}	0.0254	0.0270	0.0256	**0.0232**	0.0267	0.0239	0.0256	0.0250
Lymphography	μ_{Fit}	0.1691	0.1659	**0.1593**	0.1631	0.1694	0.1662	0.1625	0.1691
M-of-n	μ_{Fit}	0.0053	0.0048	0.0049	**0.0047**	0.0054	0.0050	0.0055	0.0050
PenglungEW	μ_{Fit}	0.3468	0.3466	0.3400	0.3465	**0.2646**	0.2920	0.3127	0.2777
SonarEW	μ_{Fit}	0.0248	0.0109	0.0177	0.0181	0.0174	0.0173	**0.0082**	0.0172
Tic-tac-toe	μ_{Fit}	**0.1544**	**0.1544**	**0.1544**	**0.1544**	**0.1544**	**0.1544**	**0.1544**	**0.1544**
Vote	μ_{Fit}	0.0022	0.0023	0.0023	0.0026	**0.0019**	**0.0019**	**0.0019**	**0.0019**
WaveformEW	μ_{Fit}	0.1596	0.1592	0.1591	**0.1563**	0.1607	0.1594	0.1590	0.1600
WineEW	μ_{Fit}	0.0032	0.0032	**0.0031**	**0.0031**	**0.0031**	**0.0031**	**0.0031**	**0.0031**
Zoo	μ_{Fit}	0.0033	0.0033	0.0034	0.0033	**0.0032**	0.0033	**0.0032**	**0.0032**
score	W	1	0	1	3	**3**	0	1	0
	T	2	2	3	3	**5**	3	5	5
	L	11	12	10	8	**6**	11	8	9

Table 8: The mean selected Features μ_{Feat} results using the eight TFs and RF with DESFO

Benchmarks	Eval-Metric	V-v1	V-v2	V-v3	V-v4	S-v1	S-v2	S-v3	S-v4
BreastCancer	μ_{Feat}	**5.000**	**5.000**	**5.000**	5.100	**5.000**	**5.000**	**5.000**	5.200
BreastEW	μ_{Feat}	12.70	12.80	12.000	12.20	10.00	9.400	11.10	**9.000**
Exactly2	μ_{Feat}	**5.000**	**5.000**	**5.000**	**5.000**	**5.000**	**5.000**	**5.000**	**5.000**
IonosphereEW	μ_{Feat}	15.10	16.00	14.500	14.60	10.50	9.000	14.30	**8.900**
KrVsKpEW	μ_{Feat}	15.50	17.90	17.700	17.10	15.90	**14.50**	15.80	15.20
Lymphography	μ_{Feat}	9.000	8.500	9.300	9.100	**7.300**	8.600	8.700	7.700
M-of-n	μ_{Feat}	6.400	6.400	6.500	**6.300**	7.000	**6.300**	6.500	6.700
PenglungEW	μ_{Feat}	156.2	150.2	142.20	155.6	43.60	81.80	105.5	**39.50**
SonarEW	μ_{Feat}	30.20	29.20	29.300	27.60	24.80	**17.40**	22.90	23.70
Tic-tac-toe	μ_{Feat}	**7.000**	**7.000**	**7.000**	**7.000**	**7.000**	**7.000**	**7.000**	**7.000**
Vote	μ_{Feat}	3.200	3.200	3.500	3.200	**2.000**	2.100	2.200	**2.000**
WaveformEW	μ_{Feat}	18.70	18.80	18.500	20.90	**16.90**	17.90	19.30	19.80
WineEW	μ_{Feat}	**3.000**	**3.000**	**3.000**	**3.000**	**3.000**	**3.000**	**3.000**	**3.000**
Zoo	μ_{Feat}	4.500	4.700	4.600	4.500	**4.000**	4.200	4.100	4.100
score	W	0	0	0	0	**1**	1	0	3
	T	4	4	4	4	**7**	6	4	4
	L	10	10	10	10	**6**	7	10	7

5.3.3 Evaluation DESFO-TFs using the SVM classifier:

Table 9 presents the mean accuracy μ_{Acc} the DESFO across eight TFs utilizing the SVM classifier. It was found that DESFO-Sv4 significantly outperformed in 10 of the 14 datasets, with DESFO-Sv1 and DESFO-Sv3 following closely behind, excelling in 9 out of the 14 datasets for both of them. As a result, when considering the mean accuracy, DESFO-Sv4 is ranked highest among the evaluated methods.

Table 10 presents the mean Fitness Value μ_{Fit} for DESFO across eight TFs, utilizing the SVM classifier. It was noted that DESFO-Sv3 demonstrated notable performance in 8 out of the 14 datasets, and DESFO-Sv4 and DESFO-Sv2 followed closely, showing significant results in 7 and 6 out of 14 datasets, respectively. Hence, when considering the mean fitness values, DESFO-Sv3 is the top-performing method among all the TFs.

Table 11 presents the mean features selected μ_{Feat} for DESFO across eight TFs, using the SVM classifier for analysis. Both DESFO-Sv2 and DESFO-Sv3 demonstrated remarkable performance in 7 out of the 14 datasets for both of them. Following them closely were DESFO-Sv1 and DESFO-Sv4, each showing significant outcomes in 5 and 6 out of the 14 datasets, respectively. Therefore, based on the average features selected, DESFO-Sv2 and DESFO-Sv3 emerge as the leading method among all the TFs considered.

Table 9: The mean Accuracy μ_{Acc} results using the eight TFs and SVM with DESFO

Benchmarks	Eval-Metric	V-v1	V-v2	V-v3	V-v4	S-v1	S-v2	S-v3	S-v4
BreastCancer	μ_{Acc}	**0.9786**	**0.9786**	**0.9786**	**0.9786**	**0.9786**	**0.9786**	**0.9786**	**0.9786**
BreastEW	μ_{Acc}	0.9474	0.9474	0.9474	0.9474	**0.9544**	0.9500	0.9491	0.9526
Exactly2	μ_{Acc}	**0.7500**	**0.7500**	**0.7500**	**0.7500**	**0.7500**	**0.7500**	**0.7500**	**0.7500**
IonosphereEW	μ_{Acc}	0.9662	0.9648	**0.9676**	0.9662	0.9648	0.9648	**0.9676**	0.9620
KrVsKpEW	μ_{Acc}	0.9820	0.9833	0.9842	0.9831	0.9847	0.9833	0.9808	**0.9850**
Lymphography	μ_{Acc}	0.8533	0.8467	0.8533	0.8533	0.8467	0.8500	**0.8633**	0.8500
M-of-n	μ_{Acc}	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
PenglungEW	μ_{Acc}	0.8200	0.8267	0.8400	0.8400	**0.9267**	0.9000	0.8733	0.9200
SonarEW	μ_{Acc}	0.9476	0.9500	0.9452	0.9429	0.9429	0.9429	0.9429	**0.9548**
Tic-tac-toe	μ_{Acc}	**0.9062**	**0.9062**	**0.9062**	**0.9062**	**0.9062**	**0.9062**	**0.9062**	**0.9062**
Vote	μ_{Acc}	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
WaveformEW	μ_{Acc}	0.8767	0.8773	0.8778	0.8768	0.8765	0.8761	0.8777	**0.8785**
WineEW	μ_{Acc}	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
Zoo	μ_{Acc}	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
score	W	0	0	0	0	2	0	1	**3**
	T	7	7	8	7	7	7	8	**7**
	L	7	7	6	7	5	7	5	**4**

Table 10: The mean Fitness value μ_{Fit} results using the eight TFs and SVM with DESFO

Benchmarks	Eval-Metric	V-v1	V-v2	V-v3	V-v4	S-v1	S-v2	S-v3	S-v4
BreastCancer	μ_{Fit}	**0.0262**	**0.0262**	**0.0262**	**0.0262**	**0.0262**	**0.0262**	**0.0262**	**0.0262**
BreastEW	μ_{Fit}	0.0548	0.0551	0.0550	0.0548	**0.0459**	0.0504	0.0518	0.0476
Exactly2	μ_{Fit}	**0.2483**	**0.2483**	**0.2483**	**0.2483**	**0.2483**	**0.2483**	**0.2483**	**0.2483**
IonosphereEW	μ_{Fit}	0.0370	0.0384	0.0361	0.0371	0.0380	0.0378	**0.0357**	0.0409
KrVsKpEW	μ_{Fit}	0.0241	0.0227	0.0220	0.0230	0.0218	0.0229	0.0246	**0.0211**
Lymphography	μ_{Fit}	0.1495	0.1557	0.1495	0.1493	0.1562	0.1525	**0.1397**	0.1530
M-of-n	μ_{Fit}	0.0051	**0.0048**	0.0050	0.0049	0.0051	0.0049	0.0050	0.0052
PenglungEW	μ_{Fit}	0.1812	0.1746	0.1616	0.1616	**0.0734**	0.1004	0.1275	0.0801
SonarEW	μ_{Fit}	0.0558	0.0536	0.0581	0.0605	0.0602	0.0598	0.0599	**0.0487**
Tic-tac-toe	μ_{Fit}	**0.1017**	**0.1017**	**0.1017**	**0.1017**	0.1018	**0.1017**	**0.1017**	**0.1017**
Vote	μ_{Fit}	0.0022	0.0026	0.0020	0.0021	**0.0019**	**0.0019**	**0.0019**	**0.0019**
WaveformEW	μ_{Fit}	0.1280	0.1271	**0.1269**	0.1281	0.1284	0.1286	0.1273	0.1272
WineEW	μ_{Fit}	0.0018	0.0018	**0.0015**	0.0018	**0.0015**	**0.0015**	**0.0015**	**0.0015**
Zoo	μ_{Fit}	0.0033	0.0033	0.0034	0.0032	0.0033	0.0032	**0.0031**	0.0032
score	W	0	1	1	0	2	0	**3**	2
	T	3	3	4	3	4	5	**5**	5
	L	11	10	9	11	8	9	**6**	7

Table 11: The mean selected Features μ_{Feat} results using the eight TFs and SVM with DESFO

Benchmarks	Eval-Metric	V-v1	V-v2	V-v3	V-v4	S-v1	S-v2	S-v3	S-v4
BreastCancer	μ_{Feat}	**5.000**	8.000	**5.000**	**5.000**	**5.000**	**5.000**	**5.000**	**5.000**
BreastEW	μ_{Feat}	8.200	3.500	8.700	8.000	2.300	2.700	4.200	**2.200**
Exactly2	μ_{Feat}	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
IonosphereEW	μ_{Feat}	12.10	11.90	13.80	12.40	10.700	**10.00**	12.40	10.90
KrVsKpEW	μ_{Feat}	22.70	22.00	22.90	22.60	24.000	22.70	**20.10**	22.40
Lymphography	μ_{Feat}	7.800	**7.100**	7.800	7.400	8.000	7.200	7.900	8.100
M-of-n	μ_{Feat}	6.600	**6.200**	6.500	6.400	6.600	6.400	6.500	6.700
PenglungEW	μ_{Feat}	97.00	98.70	103.2	102.4	**26.400**	44.600	68.20	30.30
SonarEW	μ_{Feat}	23.90	24.70	23.30	23.50	21.700	**19.50**	19.90	23.20
Tic-tac-toe	μ_{Feat}	**8.000**	**8.000**	**8.000**	**8.000**	8.100	**8.000**	**8.000**	**8.000**
Vote	μ_{Feat}	3.500	4.200	3.200	3.300	**3.000**	**3.000**	**3.000**	**3.000**
WaveformEW	μ_{Feat}	23.70	**22.50**	23.60	24.70	24.500	23.90	24.70	27.50
WineEW	μ_{Feat}	2.400	2.300	**2.000**	2.300	**2.000**	**2.000**	**2.000**	**2.000**
Zoo	μ_{Feat}	5.200	5.300	5.500	5.100	5.200	5.100	**5.000**	5.100
score	W	0	3	0	0	1	**2**	**2**	1
	T	3	2	4	3	4	**5**	**5**	5
	L	11	9	10	11	9	**7**	**7**	8

5.4 The Overall Evaluation and Discussion:

Table 12 shows the overall evaluation of the DESFO behavior with the eight TFs and ML classifiers in terms of the best accuracy, fitness value and selected features results.

Table 12: The three primary classifiers and their respective top-performing binary versions in the suggested DESFO algorithm.

Classifier	Metric	Best-performing TF	# of superiority Benchmarks
K-NN	μ_{Acc}	DESFO-V-v4	8
	μ_{Fit}	DESFO-S-v1	7
	μ_{Feat}	DESFO-S-v1 and DESFO-S-v4	7
RF	μ_{Acc}	DESFO-V-v4	10
	μ_{Fit}	DESFO-S-v1	8
	μ_{Feat}	DESFO-S-v1	8
SVM	μ_{Acc}	DESFO-S-v4	10
	μ_{Fit}	DESFO- S-v3	8
	μ_{Feat}	DESFO-S-v2 and S-v3	7

Overall, according to Table 12, the optimal combinations of the DESFO variant and three classifiers alongside eight TFs have been identified, setting the stage for subsequent experiments in this section. The models that emerged as the most effective in terms of μ_{Acc} are DESFO-Vv4–RF and DESFO-S-v4–SVM. When considering theμ_{Feat}, the most effective models identified are DESFO-S-v1 and DESFO-S-v3. Furthermore, in terms ofμ_{Feat}, DESFO-S-v1 stands out as the most effective model.

6. Conclusion and future works

In the study, the behavior of the DESFO algorithm is introduced, merging the DE and SFO algorithms with eight (TFs) divided into two categories: V-shaped and S-shaped. This merger was aimed at improving (FS) strategies. Classifiers such as K-NN, RF, and SVM were employed to assess the efficacy of the selected feature groups and determine classification accuracy. This study conducted tests across various benchmarks that featured multi-scale features and multi-records to prove its success. Results from the three classifiers were juxtaposed against the performances of the eight V-shaped and S-shaped (TFs). Evaluation metrics included mean fitness value obtained, mean accuracy obtained, and mean number of features selected. The results revealed that considering mean accuracy, the DESFO algorithm, when paired with the RF classifier and V-shaped V4 (TFs), as well as with the SVM classifier and S-shaped V4 (TFs), performed better than all other configurations on 10 out of 14 benchmarks analyzed. For mean fitness functions, the DESFO technique performed best with the RF classifier and V-shaped V1 (TFs) and with the SVM classifier and S-shaped V3, leading in 8 of the 14 benchmarks. Additionally, in terms of the mean of selected features, the DESFO algorithm, combined with the RF classifier and equipped with the S-shaped V1, was the standout performer in 8 out of the 14 benchmarks.

Further exploration into the effectiveness of DESFO for (FS), utilizing various ML algorithms like Naive Bayes, Logistic Regression, Decision Trees (DT), and more, is warranted. Due to its established prowess in selecting features, the DESFO holds significant potential across multiple domains, including Engineering, software cost estimation, the Internet of Things (IoT), networking security, healthcare and intrusion detection systems (IDS).

Conflict of interest:

The authors affirm that there are no conflicts of interest or personal connections that could seem to influence the research detailed in this paper.

Data availability:

The data sets employed in our study are stored in a publicly accessible repository tailored for ML and data classification tasks, specifically the UC Irvine ML Repository (https://archive.ics.uci.edu/datasets). It is noteworthy to mention that we utilized 14 different datasets in our research.

References

[1] R. Bellman, “Dynamic programming,” *Science (80-.).*, vol. 153, no. 3731, pp. 34–37, 1966.

[2] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh, *Feature extraction: foundations and applications*, vol. 207. Springer, 2008.

[3] B. Remeseiro and V. Bolon-Canedo, “A review of feature selection methods in medical applications,” *Comput. Biol. Med.*, vol. 112, no. February, p. 103375, 2019, doi: 10.1016/j.compbiomed.2019.103375.

[4] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, “Recent advances and emerging challenges of feature selection in the context of big data,” *Knowledge-Based Syst.*, vol. 86, pp. 33–45, 2015, doi: 10.1016/j.knosys.2015.05.014.

[5] V. F. Rodriguez-Galiano, J. A. Luque-Espinar, M. Chica-Olmo, and M. P. Mendes, “Feature selection approaches for predictive modelling of groundwater nitrate pollution: An evaluation

of filters, embedded and wrapper methods," *Sci. Total Environ.*, vol. 624, pp. 661–672, 2018, doi: 10.1016/j.scitotenv.2017.12.152.

[6] I. Guyon and A. Elisseeff, "An introduction to feature extraction," in *Feature extraction: foundations and applications*, Springer, 2006, pp. 1–25.

[7] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, no. Mar, pp. 1157–1182, 2003.

[8] T. N. Lal, O. Chapelle, J. Weston, and A. Elisseeff, "Embedded methods," in *Feature extraction: Foundations and applications*, Springer, 2006, pp. 137–165.

[9] Y. Bazi and F. Melgani, "Toward an optimal SVM classification system for hyperspectral remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 11, pp. 3374–3385, 2006.

[10] E. Y. Boateng, J. Otoo, and D. A. Abaye, "Basic Tenets of Classification Algorithms K-Nearest-Neighbor, Support Vector Machine, Random Forest and Neural Network: A Review," *J. Data Anal. Inf. Process.*, vol. 08, no. 04, pp. 341–357, 2020, doi: 10.4236/jdaip.2020.84020.

[11] W. M. Brown, T. D. Gedeon, D. I. Groves, and R. G. Barnes, "Artificial neural networks: A new method for mineral prospectivity mapping," *Aust. J. Earth Sci.*, vol. 47, no. 4, pp. 757–770, 2000, doi: 10.1046/j.1440-0952.2000.00807.x.

[12] M. Abedi, G. H. Norouzi, and A. Bahroudi, "Support vector machine for multi-classification of mineral prospectivity areas," *Comput. Geosci.*, vol. 46, pp. 272–283, 2012, doi: 10.1016/j.cageo.2011.12.014.

[13] V. F. Rodriguez-Galiano, B. Ghimire, J. Rogan, M. Chica-Olmo, and J. P. Rigol-Sanchez, "An assessment of the effectiveness of a random forest classifier for land-cover classification," *ISPRS J. Photogramm. Remote Sens.*, vol. 67, no. 1, pp. 93–104, 2012, doi: 10.1016/j.isprsjprs.2011.11.002.

[14] A. A. Zhigljavsky, "Theory of Global Random Search (Mathematics and its Applications)," 1991, [Online]. Available: http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20&path=ASIN/0792311221%5Cnhttp://www.amazon.de/exec/obidos/redirect?tag=citeulike01-21&path=ASIN/0792311221%5Cnhttp://www.amazon.fr/exec/obidos/redirect?tag=citeulike06-21&path=ASIN/07

[15] E. Amaldi and V. Kann, "On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems," *Theor. Comput. Sci.*, vol. 209, no. 1–2, pp. 237–260, 1998.

[16] R. A. Khurma, I. Aljarah, and A. Sharieh, "A Simultaneous Moth Flame Optimizer Feature Selection Approach Based on Levy Flight and Selection Operators for Medical Diagnosis," *Arab. J. Sci. Eng.*, vol. 46, no. 9, pp. 8415–8440, 2021, doi: 10.1007/s13369-021-05478-x.

[17] D. Rodrigues, X. S. Yang, A. N. De Souza, and J. P. Papa, "Binary flower pollination algorithm and its application to feature selection," *Stud. Comput. Intell.*, vol. 585, no. January, pp. 85–100, 2015, doi: 10.1007/978-3-319-13826-8_5.

[18] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: Optimization by a colony of cooperating agents," *IEEE Trans. Syst. Man, Cybern. Part B Cybern.*, vol. 26, no. 1, pp. 29–41, 1996, doi: 10.1109/3477.484436.

[19] R. Eberhart and J. K. Sixth, "A new optimizer using particle swarm theory," *Proc. IEEE*

Symp. Micro Mach. Hum. Sci. Nagoys, Japan, pp. 39–43, 1997, [Online]. Available: https://ieeexplore.ieee.org/abstract/document/494215.?casa_token=VRHbIOq0xY0AAAAA:tigoKrFPGIOWOZPL3HUCxeJDuwpHdMr7AdrNcyfXSzfY9zdeQ3AAVzx9vd-b63ZQ8Q1ZwFq8E5okfcE

[20] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," *J. Glob. Optim.*, vol. 39, no. 3, pp. 459–471, 2007, doi: 10.1007/s10898-007-9149-x.

[21] A. A. Abd El-Mageed, A. A. Abohany, and A. Elashry, "Effective Feature Selection Strategy for Supervised Classification based on an Improved Binary Aquila Optimization Algorithm," *Comput. Ind. Eng.*, vol. 181, p. 109300, 2023.

[22] M. Mafarja *et al.*, "Binary dragonfly optimization for feature selection using time-varying transfer functions," *Knowledge-Based Syst.*, vol. 161, no. December 2017, pp. 185–204, 2018, doi: 10.1016/j.knosys.2018.08.003.

[23] J. C. Bansal and K. Deep, "A modified binary particle swarm optimization for Knapsack problems," *Appl. Math. Comput.*, vol. 218, no. 22, pp. 11042–11061, 2012, doi: 10.1016/j.amc.2012.05.001.

[24] S. Mirjalili and A. Lewis, "S-shaped versus V-shaped transfer functions for binary Particle Swarm Optimization," *Swarm Evol. Comput.*, vol. 9, pp. 1–14, 2013, doi: 10.1016/j.swevo.2012.09.002.

[25] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *J. Glob. Optim.*, vol. 11, pp. 341–359, 1997.

[26] S. Shadravan, H. R. Naji, and V. K. Bardsiri, "The Sailfish Optimizer: A novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems," *Eng. Appl. Artif. Intell.*, vol. 80, no. February, pp. 20–34, 2019, doi: 10.1016/j.engappai.2019.01.001.

[27] A. G. Gad, K. M. Sallam, R. K. Chakrabortty, M. J. Ryan, and A. A. Abohany, *An improved binary sparrow search algorithm for feature selection in data classification*, vol. 34, no. 18. Springer London, 2022. doi: 10.1007/s00521-022-07203-7.

[28] I. Klyueva, "Hybrid approach to improving the results of the SVM classification using the random forest algorithm/L. Demidova, I. Klyueva, A. Pylkin," *Procedia Comput. Sci.*, vol. 150, pp. 455–461, 2019.

[29] A. Murugan, S. A. H. Nair, and K. P. S. Kumar, "Detection of skin cancer using SVM, random forest and kNN classifiers," *J. Med. Syst.*, vol. 43, pp. 1–9, 2019.

[30] H. Faris *et al.*, "An efficient binary salp swarm algorithm with crossover scheme for feature selection problems," *Knowledge-Based Syst.*, vol. 154, pp. 43–67, 2018.

[31] J. Too, A. R. Abdullah, and N. Mohd Saad, "A new quadratic binary harris hawk optimization for feature selection," *Electronics*, vol. 8, no. 10, p. 1130, 2019.

[32] M. M. Mafarja and S. Mirjalili, "Hybrid binary ant lion optimizer with rough set and approximate entropy reducts for feature selection," *Soft Comput.*, vol. 23, no. 15, pp. 6249–6265, 2019.

[33] Y. Chen *et al.*, "A Hybrid Binary Dragonfly Algorithm with an Adaptive Directed Differential Operator for Feature Selection," *Remote Sens.*, vol. 15, no. 16, p. 3980, 2023.

[34] H. Chantar, M. Mafarja, H. Alsawalqah, A. A. Heidari, I. Aljarah, and H. Faris, "Feature

selection using binary grey wolf optimizer with elite-based crossover for Arabic text classification," *Neural Comput. Appl.*, vol. 32, pp. 12201–12220, 2020.

[35] R. B. Prudhvi, K. V Sai, Y. Abhishek, and K. Venkatanareshbabu, "Feature selection using binary psogsa and radial basis network with a novel fitness function," in *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, IEEE, 2018, pp. 1–6.

[36] A. G. Hussien, A. E. Hassanien, E. H. Houssein, S. Bhattacharyya, and M. Amin, *S-shaped binary whale optimization algorithm for feature selection*, vol. 727, no. June 2018. Springer Singapore, 2019. doi: 10.1007/978-981-10-8863-6_9.

[37] A. Fatahi, M. H. Nadimi-Shahraki, and H. Zamani, "An improved binary quantum-based avian navigation optimizer algorithm to select effective feature subset from medical data: A COVID-19 case study," *J. Bionic Eng.*, vol. 21, no. 1, pp. 426–446, 2024.

[38] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, 1997.

[39] S. M. Azzam, O. E. Emam, and A. S. Abolaban, "An improved Differential evolution with Sailfish optimizer (DESFO) for handling feature selection problem," *Sci. Rep.*, vol. 14, no. 1, p. 13517, 2024.

[40] K. M. Sallam, S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Multi-method based orthogonal experimental design algorithm for solving CEC2017 competition problems," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, 2017, pp. 1350–1357.

[41] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. theory*, vol. 13, no. 1, pp. 21–27, 1967.

[42] Y. Wu, K. Ianakiev, and V. Govindaraju, "Improved k-nearest neighbor classification," *Pattern Recognit.*, vol. 35, no. 10, pp. 2311–2318, 2002.

[43] M. Mafarja and S. Mirjalili, "Whale optimization approaches for wrapper feature selection," *Appl. Soft Comput.*, vol. 62, pp. 441–453, 2018.

[44] M. J. Zaki and W. Meira, *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press, 2014.

[45] R. Katuwal, P. N. Suganthan, and L. Zhang, "An ensemble of decision trees with random vector functional link networks for multi-class classification," *Appl. Soft Comput.*, vol. 70, pp. 1146–1153, 2018.

[46] H. Cao, S. Bernard, R. Sabourin, and L. Heutte, "Random forest dissimilarity based multi-view learning for radiomics application," *Pattern Recognit.*, vol. 88, pp. 185–197, 2019.

[47] A. Criminisi, J. Shotton, and E. Konukoglu, "Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning," *Found. trends® Comput. Graph. Vis.*, vol. 7, no. 2–3, pp. 81–227, 2012.

[48] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 2013.

[49] A. Tharwat, A. E. Hassanien, and B. E. Elnaghi, "A BA-based algorithm for parameter optimization of support vector machine," *Pattern Recognit. Lett.*, vol. 93, pp. 13–22, 2017.

[50] A. Frank, "UCI machine learning repository," *http//archive. ics. uci. edu/ml*, 2010.