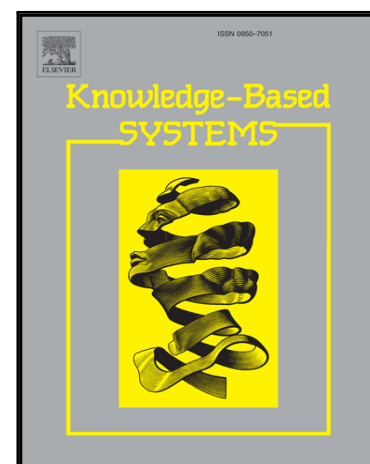# Accepted Manuscript

Binary Dragonfly Optimization for Feature Selection using Time-Varying Transfer functions

Majdi Mafarja, Ibrahim Aljarah, Ali Asghar Heidari, Hossam Faris, Philippe Fournier-Viger, Xiaodong Li, Seyedali Mirjalili

Please cite this article as: Majdi Mafarja, Ibrahim Aljarah, Ali Asghar Heidari, Hossam Faris, Philippe Fournier-Viger, Xiaodong Li, Seyedali Mirjalili, Binary Dragonfly Optimization for Feature Selection using Time-Varying Transfer functions, *Knowledge-Based Systems* (2018), doi: https://doi.org/10.1016/j.knosys.2018.08.003

Highlights

- Novel feature selection approaches based on Binary Dragonfly Algorithm (BDA) are proposed.

- Eight time varying S-shaped and V-shaped transfer functions are proposed.

- The leverage of using time-varying transfer functions on exploration and exploitation behaviors is investigated.

- Extensive tests are made to assess the proposed algorithms on the datasets to prove their merits
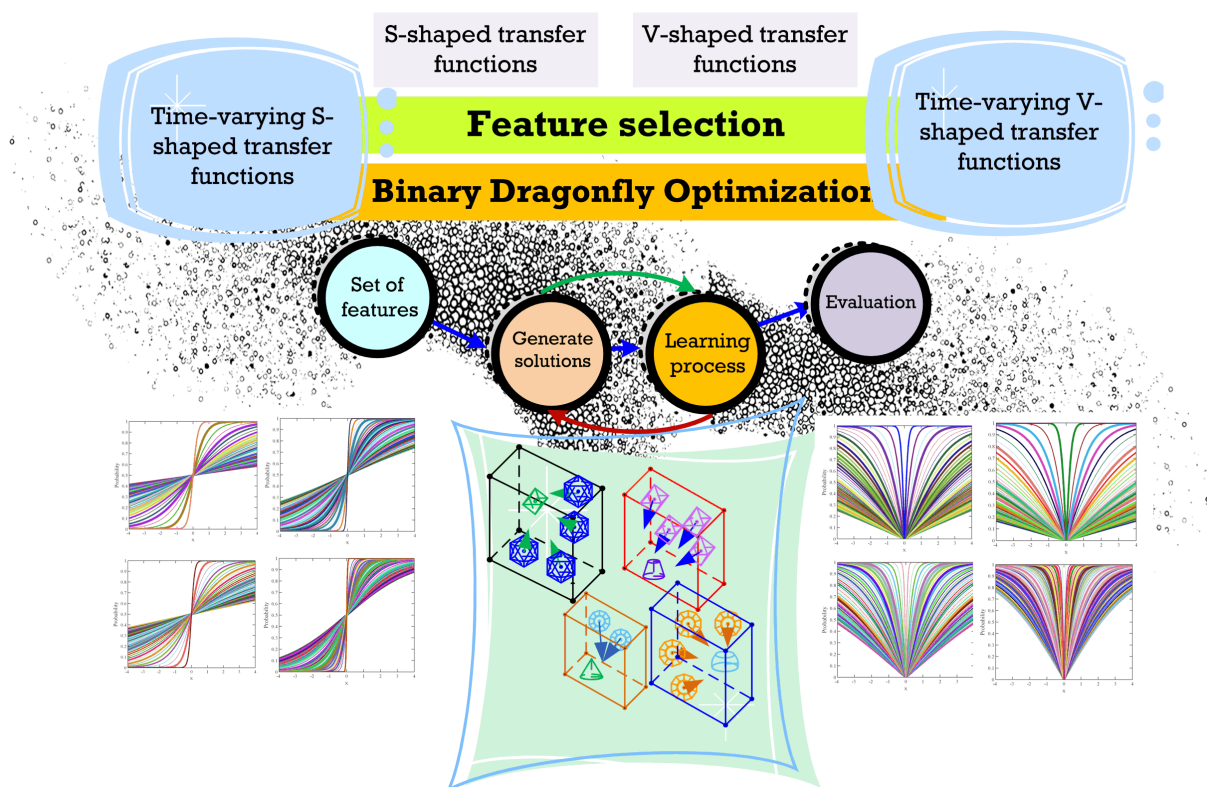
Binary Dragonfly Optimization for Feature Selection using Time-Varying Transfer functions

Majdi Mafarja[a], Ibrahim Aljarah[b], Ali Asghar Heidari[c], Hossam Faris[b], Philippe Fournier-Viger[d], Xiaodong Li[e], Seyedali Mirjalili[f]

[a]*Department of Computer Science, Birzeit University, Birzeit, Palestine*
mmafarja@birzeit.edu, mmafarjeh@gmail.com
[b]*King Abdullah II School for Information Technology, The University of Jordan, Amman, Jordan*
{hossam.faris,i.aljarah}@ju.edu.jo
[c]*School of Surveying and Geospatial Engineering, University of Tehran, Tehran, Iran*
as_heidari@ut.ac.ir
[d]*School of Natural Sciences and Humanities*
Harbin Institute of Technology (Shenzhen), HIT Campus, Xili, Shenzhen, China
philfv@hitsz.edu.cn
[e]*School of Science, RMIT University, Melbourne, Australia*
xiaodong.li@rmit.edu.au
[f]*Institute of Integrated and Intelligent Systems, Griffith University, Nathan, Brisbane, QLD 4111, Australia*
seyedali.mirjalili@griffithuni.edu.au

Abstract

The Dragonfly Algorithm (DA) is a recently proposed heuristic search algorithm that was shown to have excellent performance for numerous optimization problems. In this paper, a wrapper-feature selection algorithm is proposed based on the Binary Dragonfly Algorithm (BDA). The key component of the BDA is the transfer function that maps a continuous search space to a discrete search space. In this study, eight transfer functions, categorized into two families (S-shaped and V-shaped functions) are integrated into the BDA and evaluated using eighteen benchmark datasets obtained from the UCI data repository. The main contribution of this paper is the proposal of time-varying S-shaped and V-shaped transfer functions to leverage the impact of the step vector on balancing exploration and exploitation. During the early stages of the optimization process, the probability of changing the position of an element is high, which facilitates the exploration of new solutions starting from the initial population. On the other hand, the probability of changing the position of an element becomes lower towards the end of the optimization process. This behavior is obtained by considering the current iteration number as a parameter of transfer functions. The performance of the proposed approaches is compared with that of other state-of-art approaches including the DA, binary grey wolf optimizer (bGWO), binary gravitational search algorithm (BGSA), binary bat algorithm (BBA), particle swarm optimization (PSO), and genetic algorithm in terms of classification accuracy, sensitivity, specificity, area under the curve, and number of selected attributes. Results show that the time-varying S-shaped BDA approach outperforms compared approaches.

1. Introduction

In the past decades, metaheuristic and evolutionary algorithms were shown to be very successful for solving various optimization problems [1, 2, 3, 4, 5]. The Dragonfly Algorithm (DA) is a recent metaheuristic, which is inspired by the behavior of dragonflies [6]. The DA can be regarded as a recently successful algorithm that can outperform other well-regarded optimizers. The DA has been applied to several real-world problems such as economic emission dispatch in power systems [7, 8], simulation building [9], wireless node localization in computer networks [10] and machine learning [11, 12]. The DA has shown an excellent performance for several continuous, discrete, single-objective and multi-objective optimization problems compared to several state-of-the-art metaheuristic and evolutionary algorithms such as Particle Swarm Optimization (PSO) and Differential Evolution (DE).

Up to 2018, several works have utilized the DA or improved its performance to tackle practical tasks such as photovoltaic systems [13], extension of RFID network lifetime [14], 0-1 knapsack problems [15], and economic emission dispatch problem [16]. In 2017, KS and Murugan [17] proposed a memory-based hybrid DA that integrates concepts of PSO for dealing with global optimization cases. Song and Li [18] proposed a modified DA with elite opposition learning for global optimization.

Recently, a binary version of the DA called BDA was proposed by Mirjalili [6], which applies a transfer function (TF) to map a continuous search space to a discrete one. The potential of BDA was initially evaluated on some feature selection problems and results have shown that this method has acceptable performance [19].

In general, a TF must map a continuous search space to a discrete space. In this regard, selecting a suitable TF is an important decision for improving the performance of binary velocity-based algorithms (*e.g.* PSO and DA) [20, 21]. Using TFs is recommended in several works due to several reasons [20, 22]. Firstly, TFs are algorithm independent and do not impact the search behavior of an algorithm. Secondly, the computational complexity of the algorithm does not change since the TF is calculated for each solution during each iteration. Thirdly, exploration and exploitation can be boosted when using a TF. The core drawback of a TF comes from the nature of this component. Transfer functions map velocity to probability. Hence, an algorithm should have a velocity vector. In other words, it is not a generic operator that can be used for all optimization algorithms. The other drawback of typically employed TFs within binary optimizers (*e.g.* the sigmoid, tangent and log-sigmoid functions [21, 20, 23]) is that they do not adapt exploration and exploitation in an evolutionary way during the search for solutions. They calculate the probability of changing the value of parameters in a non-adaptive way.

The operators of population-based algorithms are the only components that guide exploration and exploitation. Improving the performance of any population-based optimizer such as the DA requires to select an appropriate balance between exploration (diversification) and exploitation (intensification). Generally speaking, exploration is more important

4

than intensification during the early stages of the selection process, to explore promising regions of the feature space. But during the later stages, exploitation becomes more important because we need to increase the probability of discovering better solutions, close to those found in the previous phases. To solve challenging and high-dimensional problems, where a good balance between exploration and exploitation is required, hybrid methods can be developed. However, such techniques normally increase the computational cost of the whole optimization process.

Based on the aforementioned observations and since DA utilizes velocity vectors for updating solutions, the main contribution of this paper is to propose several time-varying TFs for the DA for binary tasks. This work demonstrates and argues that how a TF can be utilized to fine-tune and control the exploration and exploitation behavior of an optimizer is important. In other words, a TF can play a key role in tuning the exploration and exploitation phases of an optimizer rather than just converting a continuous search space to a binary one.

Based on this idea, this study proposes the use of a controlling parameter that has a gradually decreasing influence over the course of iterations, and that idea is applied to BDA. To assess the performance of BDA, this paper considers the task of feature selection, as it is a fundamental binary optimization problem that is challenging and high-dimensional, and has many applications.

This work proposes a wrapper feature selection approach that utilizes the recent BDA as a search strategy and the K-Nearest Neighborhood (KNN) classifier as an evaluator. The main aim of this study is to propose and investigate the effects of eight time-dependent TFs on the efficiency of BDA. Moreover, the results from the eight proposed approaches are compared with the original DA and some other state-of-the-art algorithms.

The contributions of this research can be summarized as follows:

- Several binary variants of the DA are proposed.

- This paper investigates the exploration and exploitation behavior of BDA with existing TFs and identifies their possible negative effects on the balance between exploration and exploitation.

- Various time-dependent S-shaped and V-shaped TFs are proposed to effectively overcome the drawbacks of existing TFs, and provide a stable balance between exploration and exploitation in BDA.

- The improved searching capabilities of the proposed time-dependent variants of the BDA is evaluated on several well-regarded feature selection datasets, and excellent results are obtained.

The rest of the paper is organized as follows. A theoretical background about feature selection is presented and related work is reviewed in Section 2. Section 3 describes the DA. Section 4 introduces the BDA. Section 5 presents the proposed approach. Experimental results and discussion are presented in Section 6, and finally, a conclusion is drawn and opportunities for future work are discussed in Section 7.

2. Background on feature selection

Feature selection (FS) plays an important role in various machine learning and data mining tasks such as object-based image classification [24], prediction of groundwater nitrate pollution [25], intrusion detection [26], automatic satire and irony detection [27], music streaming recommendation [28], spam detection [29], financial distress prediction [30], and classification [31]. Bolón-Canedo et al. [32] discussed challenges of FS in the context of big data. FS aims at improving classification accuracy by eliminating redundant, irrelevant, and noisy data from a dataset. According to Liu and Motoda [33], FS algorithms can be classified based on two main criteria: their subset evaluation procedure and their searching procedure. In terms of the former, FS methods are categorized as filters and wrappers [33]. Wrapper approaches utilizes a learning algorithm (*e.g.* classifier) to evaluate feature subsets, while filters evaluate a feature subset using the data itself (*e.g.* using a measure such as the information gain) [34].

Finding an optimal subset of features for FS problems is challenging due to the large number of possible combinations. A naive approach to solve a FS problem is to apply a brute-force search and generate all possible subsets of features to find the best one. If the original dataset contains k features, then there are 2^k-1 subsets to be generated and evaluated.

Due to the exponential complexity of this approach, it is impractical when N is very large. A more practical solution to solve FS problems is to utilize a heuristic search [35]. As the name suggests, the search is guided using heuristic information collected during the optimization process. Although heuristic search techniques do not guarantee finding the best subset of features, they can generally produce an acceptable solution quickly [36].

Metaheuristics are general purpose algorithms, which have been readily applied to a wide range of problems [36]. Nature-inspired algorithms are mostly metaheuristics and mimic the social and biological behaviors of creatures in nature. Various nature-inspired algorithms have been utilized to tackle the FS problem in the literature such as GA [37], PSO [38, 39, 40, 41], Ant Colony Optimization (ACO) [42], DE [43], Bacterial Foraging Optimization (BFO) [44], and Artificial Bee Colony (ABC) [45].

Recently, new nature-inspired algorithms have been proposed and have shown improved results for the FS problems. For instance, an Ant Lion Optimizer (ALO) [46], which mimics the hunting behavior of antlions, has been employed as a wrapper FS method [47, 48]. Grey Wolf Optimizer (GWO) is another recent algorithm [49, 50, 51] that has been successfully employed for solving feature selection problems [52, 53]. Moth-Flame Optimizer (MFO) [54] is an algorithm that mimics the navigation method of moths and has been applied to the FS problem by Zawbaa et al. [55]. Multi-Verse Optimizer (MVO) is another recent example of metaheuristics that was applied in combination with classifiers to some FS problems Faris et al. [56].

Mafarja and Abdullah [57] proposed a mimetic filter FS approach that combines the capability of Simulated Annealing (SA) as a local search algorithm with a GA.

In subsequent work in [58], SA was hybridized with the Whale Optimization Algorithm (WOA) to form a wrapper FS approach. WOA was also recently used as a wrapper FS approach in [59]. In that approach, evolutionary operators (i.e., crossover, mutation

and selection) were employed to enhance both the exploration and exploitation capabilities of the WOA. The reported results revealed the benefits of combining those operators with WOA. In [48], the performance of the Ant Lion Optimizer (ALO) algorithm with eight different TFs was investigated. Another study Faris et al. [60] proposed a novel FS approach based on a recent metaheuristic algorithm called Multi-Verse Optimizer (MVO). In 2018, Mafarja et al. [61] proposed an improved grasshopper optimization algorithm (GOA) with new evolutionary-based operators, which is called GOA-EPD, to develop an efficient wrapper FS method. Salp Swarm Algorithm (SSA) is another recent metaheuristic that has been used in a wrapper FS method in [62].

3. Overview of the Dragonfly Algorithm

The Dragonfly Algorithm is a recently proposed swarm-based algorithm [6]. The DA mimics the hunting and migration mechanisms of idealized dragonflies. The hunting mechanism is called static swarm (feeding), in which the dragonflies fly in small groups over a small area to search for food sources. The migration mechanism is called dynamic swarm (migratory). In this phase, the dragonflies fly along one direction in larger groups so that the swarm migrates. Static and dynamic swarms are illustrated in Fig 1. Similarly to other nature-inspired algorithms, the DA consists of two phases: exploration, inspired by the static swarming behavior, and exploitation, inspired by the dynamic swarming behavior.

(a) Static swarm   (b) Dynamic swarm
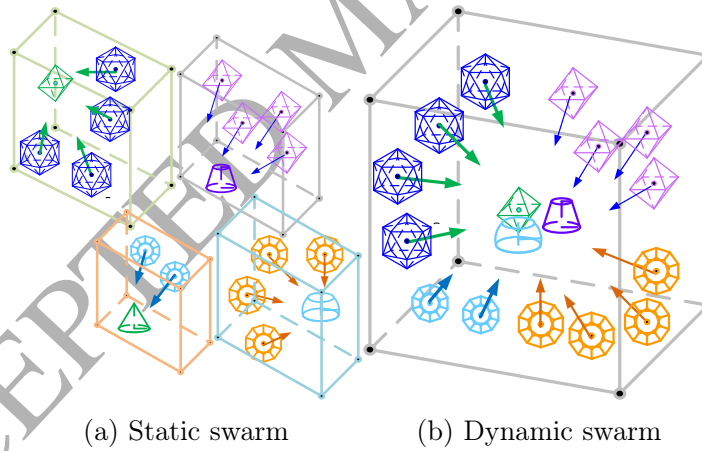
Figure 1: Static and dynamic swarming behaviors of dragonflies when foraging (each geometric figure represents a class of search agents)

To model the swarming behavior of dragonflies, five individual behaviors are utilized as follows. In the following equations, X represents the position of the current search agent, X_j represents the j-th neighbor of the X search agent, and N is the neighborhood size [63]:

- *Separation* is a mechanism that a search agent applies to stay away from other

7

neighboring search agents. This behavior is mathematically modeled as Eq. (1):

$$S_i = -\sum_{j=1}^{N} X - X_i \qquad (1)$$

- *Alignment* indicates how an individual matches its velocity with the velocity of other neighboring individuals. This behavior is mathematically modeled as Eq. (2):

$$A_i = \frac{\sum_{j=1}^{N} V_j}{N} \qquad (2)$$

where V_j represents the velocity of the j-th neighbor.

- *Cohesion* refers to the tendency of individuals to fly towards the neighboring center of mass. This behavior is mathematically modeled as Eq. (3):

$$C_i = \frac{\sum_{j=1}^{N} x_j}{N} - X \qquad (3)$$

- *Attraction* refers to the tendency of individuals to fly towards the food source. The attraction between the food source and the ith solution is mathematically modeled as Eq. (4):

$$F_i = F_{loc} - X \qquad (4)$$

where F_{loc} represents the position of the food source.

- *Distraction* refers to the tendency of individuals to fly away from an enemy. The distraction between the enemy and the ith solution is mathematically modeled as Eq. (5):

$$E_i = E_{loc} + X \qquad (5)$$

where E_{loc} represents the enemy's position.

In the DA, the food source fitness and location are supposed to be updated using the best candidate (search agent) so far. In addition, the fitness and location of the enemy should be updated using the worst candidate. This causes convergence towards promising areas and divergence outwards non-promising regions of the search space.

Based on the framework of the PSO algorithm, the DA uses two vectors to update the position of a dragonfly: the step vector (ΔX) that is similar to the velocity vector in PSO and the position vector. The step vector represents the dragonflies' movement direction. The step vector is modeled as Eq. (6):

$$\Delta X_{t+1} = (sS_i + aA_i + cC_i + fF_i + eE_i) + wX_t \qquad (6)$$

where s, w, a, c, f, and e represent the weights of the separation S_i, alignment

8

A_i, cohesion C_i, attraction towards the food source F_i, and distraction from the enemy E_i of the i-th individual's respectively. These weights enable the DA to achieve different exploration and intensification behaviors during optimization. An extensive analysis of the effect of those parameters on the DA and their values can be found in [6].

The position of an individual is updated as in Eq. (7):

$$X_{t+1} = X_t + \Delta X_{t+1} \tag{7}$$

where t is the current iteration.

Algorithm 1 shows the pseudocode of the DA. The algorithm starts by creating a random initial population. The positions and step vectors of dragonflies are randomly defined. In each iteration, the algorithm repeatedly executes the following steps until a termination criterion is satisfied. Firstly, each individual in the population is evaluated using a fitness function. Secondly, the main coefficients are updated. Thirdly, the separation (S), alignment (A), and cohesion (C), food source (F) and enemy (E) are updated using Eqs. 1 to 5. Finally, the step vectors and the position are updated using Eqs 6 and Eq. 7, respectively.

Finally, the best solution found so far is returned.

Algorithm 1 Pseudocode of the DA

Initialize the population $X_i(i = 1, 2, \ldots, n)$
Initialize $\Delta X_i(i = 1, 2, \ldots, n)$
while (end condition is not satisfied) **do**
    Evaluate each dragonfly
    Update (F) and (E)
    Update the main coefficients ($i., e., w, s, a, c, f, and\ e$)
    Calculate $S, A, C, F,$ and E (using Eqs. (1 to 5))
    Update step vectors (ΔX_{t+1}) using Eq. (6)
    Update X_{t+1} using Eq. (7)
Return the best solution

---

4. Binary Dragonfly Algorithm (BDA)

In a binary optimization problem, the search space is considered as a hypercube, where an individual can change its position from one location to another by changing one or more bits of its position vector $x = \{x_1, x_2, ..., x_d\}$. Since the original DA was designed for handling continuous optimization problems, the position of an individual is updated by adding the current position vector to the step vector. However, this mechanism cannot be used to handle a binary optimization problem such as feature selection. According to a previous study Mirjalili and Lewis [20], employing a transfer function is an effective and convenient way of converting a continuous algorithm into a binary one. Transfer functions

9

are categorized in two families according to their shapes: S-shaped and V-shaped. Fig. 2 depicts these two families of transfer functions.

(a) S-shaped TF   (b) V-shaped TF
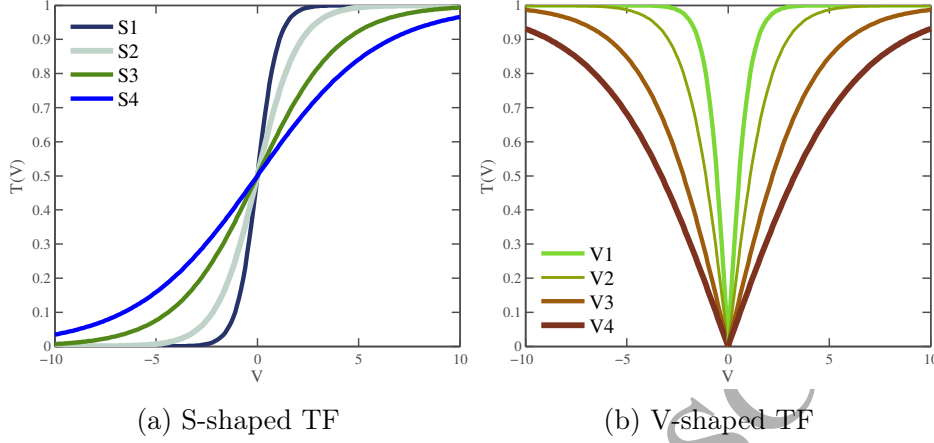
Figure 2: Transfer functions families (a) S-shaped and (b) V-shaped [20]

Generally speaking, transfer functions are used to generate the probability of changing a position's elements to 0 or 1 based on the value of the step vector (velocity) of the ith search agent in the dth dimension in the current iteration (t) as an input parameter. In a previous study [6], the transfer function of Eq. (8 was employed to calculate the probability of changing the continuous positions to binary.

$$T(v_d^i(t)) = |(v_d^i(t))/\sqrt{1 + (v_d^i(t))^2}| \tag{8}$$

The result $T(v_k^i(t))$, obtained from Eq. (8) is then used to convert the i-th element of the position vector to 0 or 1 according to Eq. 9

$$X(t+1) = \begin{cases} \neg X_t & r < T(v_k^i(t)) \\ X_t & r \geq T(v_k^i(t)) \end{cases} \tag{9}$$

where r is a random number in the [0,1] interval.

The step vector indicates the momentum of the current individual and defines the magnitude of movement. A lower step vector value indicates that the individual is very close to the best solution and needs to move in smaller steps (exploitation). In contrast, if the step vector value is large, the search agent is far from the best solution so far and requires abrupt changes (exploration) [64]. In a binary algorithm, where one uses the step vector to calculate the probability of changing positions, TFs significantly impact the balance between exploration and exploitation. If the transfer function does not change, probability will be calculated in the same manner throughout the optimization process. Changing the TF has the potential to leverage the impact of the step vector on position updates for exploring and exploiting the search space.

10

5. The proposed approach

The first TF Eq. (10) was proposed by Kennedy and Eberhart [38]. This function belongs to the S-shaped family.

$$T(v_i^k(t)) = \frac{1}{1 + e^{-v_i^k(t)}} \tag{10}$$

where $v_k^i(t)$ represents the step vector of the i-th search agent at the t-th iteration in the k-th dimension.

Each element of the vector representing the current individual will be changed according to Eq. 11 based on the probability value $T(v_k^i(t))$ obtained from Eq. 10.

$$x_i^k(t+1) = \begin{cases} 1 & rand < T(v_i^k(t+1)) \\ 0 & rand \geq T(v_i^k(t+1)) \end{cases} \tag{11}$$

An extensive study of the influence of TFs on the efficiency and final results of binary PSO was recently presented in [22]. The study evaluated three classes of TFs including S-shaped, V-shaped and linear normalized TFs.

By observing the sigmoid function in Eq. 10, it can be seen that the current version of this function does not provide a good balance between exploration and exploitation, where the exploration rate should be more than the exploitation rate at the beginning of the optimization process. Thus, some promising areas inside the search space may remain unexplored. As a result, there is a high possibility that the optimizer will get trapped in local optima (LO). A similar phenomenon can also be observed during the exploitation stage.

Inspecting the ordinary TFs (see Fig. 2), it can be seen that with the small absolute values of inputs, the probability of flipping an element in the *dth* dimension of a solution is high, while with the large absolute values of inputs the flipping probability is low. Therefore, to better explore the search space at the early stages of the optimization process, a small absolute value of step vector (close to zero) is preferable. On the other hand, to better exploit the search space in the final stages of the optimization process, a large absolute value of step vector is preferable. With the current form of the sigmoid function, a high flipping probability is produced when a low probability is required.

As a result, there is a high probability that the optimizer will oversight good solutions in the neighborhood of the solutions that have been found during the exploration phase. All these issues are caused by the fact that a TF cannot map step vector values to appropriate probability values without changing its shape during the optimization process.

Another TF that belongs to the V-shaped family (V_{T1}) is represented in Eq. 12:

$$T(\Delta x) = \begin{cases} 1 - \frac{2}{1 + e^{-2x}} & x \leq 0 \\ \frac{2}{1 + e^{-2x}} - 1 & x > 0 \end{cases} \tag{12}$$

The position of the current search agent is changed in Eq. (13) according to the probability value $T(v_k^i(t))$ obtained from Eq. 12.

$$X_{id}^{k+1} = \begin{cases} 0 & r \leq T(\Delta x_{t+1}) and \Delta x_{t+1} \leq 0 \\ 1 & r \leq T(\Delta x_{t+1}) and \Delta x_{t+1} \geq 0 \\ X_{id}^k & r > T(\Delta x_{t+1}) \end{cases} \tag{13}$$

where r is a random number in the [0,1] interval.

After examining V_{T1}, one can see that the algorithm still suffers from biased stability between the exploration and exploitation phases due to problems similar to those of the S-shaped function. Inspired by [22], an effective strategy to mitigate this problem in the BDA is to update the S-shaped TF in Eq. 10 and design a new model of TFs as shown in Eq. 14. In the proposed TFs, a time-varying scheme is employed to achieve a better balance between exploration and exploitation.

$$T(v_i^k(t), \tau) = \frac{1}{1 + e^{\frac{-v_i^k(t)}{\tau}}} \tag{14}$$

where τ is a time-varying variable that starts with an initial value and gradually decreases over iterations as shown in Eq. 15.

$$\tau = \left(1 - \frac{t}{T}\right)\tau_{max} + \frac{t}{T}\tau_{min} \tag{15}$$

where τ_{min} and τ_{max} are the min and max values of the control parameter τ, T represents the maximum number of iterations and t_{k+1} represents the current iteration.

The key fact about the newly proposed time-dependent TF is that its value can be linearly increased as the step vector of search agents increases. During early stages (when $\tau = \tau_{max}$), the probability of changing the position's element is higher, which provides higher exploration capacities for the initial population. On the other hand, the probability of changing the position's element becomes very low once $\tau = \tau_{min}$, which provides steadier exploitation inclinations in latter stages of the run.

In previous work [22], only one TF was employed and tested on the 0-1 knapsack problem, which was the improved sigmoid function. Authors of that study stated that the V-shaped TF suffers from the same problem as the S-shaped TF, but they did not study the possibility of adaptively updating this function over the iterations. In this work, the original TF is modified by adding a time-varying parameter as shown in Eq. 16. The proposed TF satisfies the design considerations mentioned in [22, 20].

$$T(\Delta x, \tau) = \begin{cases} 1 - \frac{2}{1 + e^{\frac{-2x}{\tau}}} & x \leq 0 \\ \frac{2}{1 + e^{\frac{-2x}{\tau}}} - 1 & x > 0 \end{cases} \tag{16}$$

In this work, seven new TFs manipulating the coefficient of x are introduced.

Table 1 contains the mathematical formulations of the TFs proposed in this research. Figure 3 and Figure 4 show the behaviors of the proposed time-dependent TFs.

Algorithm 2 shows the pseudocode of BDA. The algorithm starts by creating a random initial population; the position and step vectors of dragonflies are randomly defined. In each iteration, the algorithm repeatedly executes the following steps until a termination

12

Table 1: TV_S-shaped and TV_V-shaped transfer functions

TV_S family		TV_V family	
Name	Transfer function	Name	Transfer function
TV_{S1}	$T(x,\tau) = \frac{1}{1+e^{\frac{-2x}{\tau}}}$	TV_{V1}	$T(x,\tau) = \begin{cases} 1 - \frac{2}{1+e^{\frac{-2x}{\tau}}} & x \leq 0 \\ \frac{2}{1+e^{\frac{-2x}{\tau}}} - 1 & x > 0 \end{cases}$
TV_{S2}	$T(x,\tau) = \frac{1}{1+e^{\frac{-x}{\tau}}}$	TV_{V2}	$T(x,\tau) = \begin{cases} 1 - \frac{2}{1+e^{\frac{-x}{\tau}}} & x \leq 0 \\ \frac{2}{1+e^{\frac{-x}{\tau}}} - 1 & x > 0 \end{cases}$
TV_{S3}	$T(x,\tau) = \frac{1}{1+e^{\frac{-x}{2\tau}}}$	TV_{V3}	$T(x,\tau) = \begin{cases} 1 - \frac{2}{1+e^{\frac{-x}{2\tau}}} & x \leq 0 \\ \frac{2}{1+e^{\frac{-x}{2\tau}}} - 1 & x > 0 \end{cases}$
TV_{S4}	$T(x,\tau) = \frac{1}{1+e^{\frac{-x}{3\tau}}}$	TV_{V4}	$T(x,\tau) = \begin{cases} 1 - \frac{2}{1+e^{\frac{-x}{3\tau}}} & x \leq 0 \\ \frac{2}{1+e^{\frac{-x}{3\tau}}} - 1 & x > 0 \end{cases}$
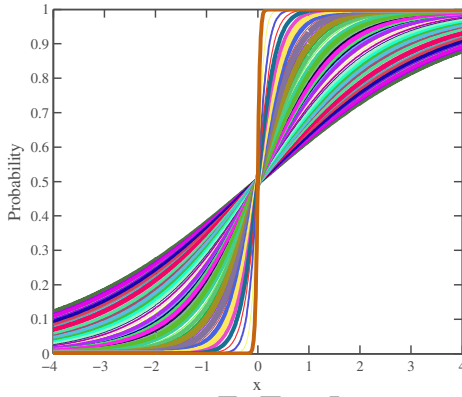
(a) TV_{S1}
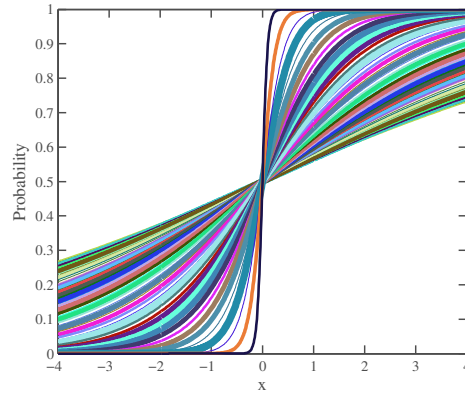
(b) TV_{S2}
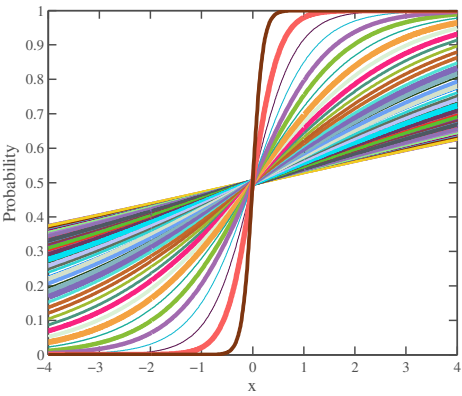
(c) TV_{S3}
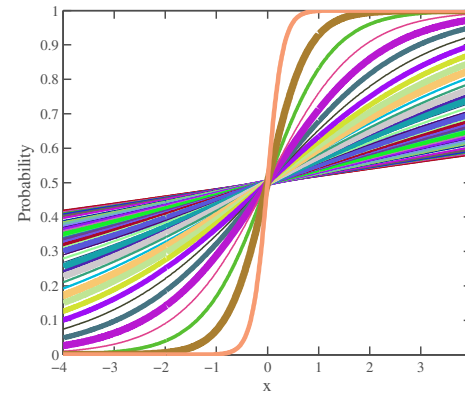
(d) TV_{S4}

Figure 3: Demonstration of the time-varying S-shaped TFs when $\tau_{max} = 4$ and $\tau_{min} = 0.01$ during 100 iterations with time step 2. Note that more vertical curves belong to the lower values of τ.

13

(a) TV_{V1}

(b) TV_{V2}

(c) TV_{V3}

(d) TV_{V4}
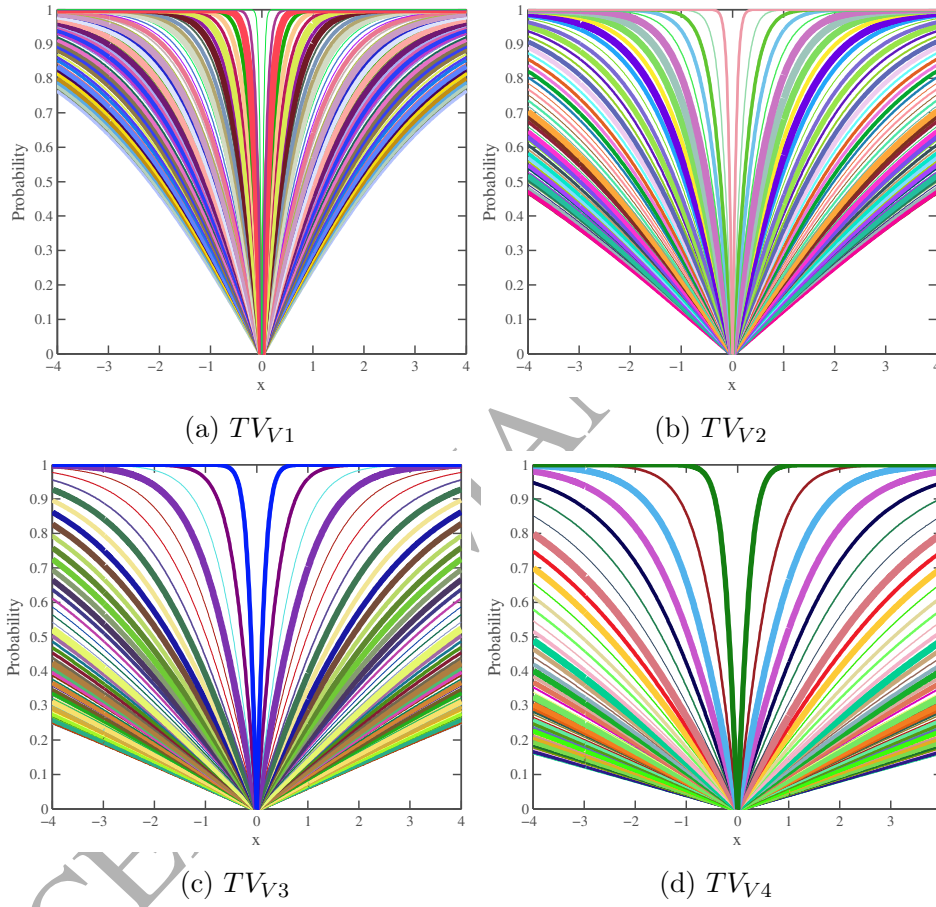
Figure 4: Demonstration of the time-varying V-shaped TFs when τ_{max} =4 and τ_{min} =0.01 during 100 iterations with time step 2. Note that more vertical curves belong to the lower values of τ.

14

criterion is satisfied. Firstly, each individual in the population is evaluated using a fitness function. Secondly, the main coefficients are updated. Thirdly, the separation (S), alignment (A), and cohesion (C), food source (F) and enemy (E) are calculated using Eqs. 1 to 5. Finally, the step vectors, transfer and the position are updated using Eqs 6, 10, and 11.

Note that adding the time-varying S-shaped and V-shaped TFs does not change the computational complexity of the DA since this component is run only once for each solution and in each iteration. Therefore, the computational complexity of binary DA is of $O(tnd)$ where t is the number of iterations, n shows the number of solutions, and d indicates the number of dimensions (variables). This complexity is identical to those of other comparative algorithms in this work: PSO, bGWO, GA, BGSA, and BBA.

Algorithm 2 Pseudocode of the BDA algorithm

Initialize the population $X_i(i = 1, 2, \ldots, n)$
Initialize $\Delta X_i(i = 1, 2, \ldots, n)$
Set τ_{max} and τ_{min}
Initialize τ using Eq. 15
while (end condition is not satisfied) **do**
    Evaluate each dragonfly
    Update (F) and (E)
    Update the main coefficients $(i., e., w, s, a, c, f, and e)$
    Calculate S, A, C, F, and E (using Eqs. (1 to 5))
    Update step vectors using Eq. (6)
    Calculate $T(\Delta X)$ using an equation from Table 1
    Update X_{t+1} using Eq. 11 or Eq. 13
Return the best solution

5.1. BDA for feature selection

Feature selection is considered as a binary optimization problem, where solutions are restricted to binary values. Therefore, the binary version of the DA could be employed to solve this problem. In this work, a vector of zeros and ones is used to represent a solution to a FS problem, where a zero indicates that the corresponding feature is not selected and a one indicates that the corresponding feature is selected. The length of the solution vector is equal to the number of features in the original dataset. In this work, eight wrapper feature selection approaches using the BDA are proposed. Each approach utilizes a transfer function to convert a continuous value to a binary value. The KNN classifier [65] evaluates the selected feature subsets. The fitness function considers classification accuracy and the number of selected features, which fulfill the consideration that FS is a multi-objective problem. The objective function is presented in Eq. 17:

$$\downarrow Fitness = \alpha \gamma_R(D) + \beta \frac{|C|}{|N|} \tag{17}$$

where $\gamma_R(D)$ represents the classification error rate, while $|C|$ is the number of selected features and $|N|$ is the total number of features in the original dataset, α and β are two parameters corresponding to the importance of classification quality and subset length, α is in the [0,1] interval and $\beta = (1 - \alpha)$ is adapted from [59].

6. Experimental evaluation and discussion

6.1. Experimental setup

The proposed approaches were tested on eighteen feature selection datasets obtained from the UCI data repository [66]. Table 2 shows characteristics of these datasets. The datasets belong to different fields (*e.g.* biology, politics, games, physics, and chemistry) and are of different sizes. The diversity of the selected datasets facilitates benchmarking of the proposed approaches from different perspectives. In the proposed approach, to validate the optimality of the results, the hold-out strategy was used, where each dataset is randomly divided into two parts; training and testing, where 80% of each dataset is used for training and the remaining 20% is for testing purposes [59]. All experiments were repeated for 30 independent times to obtain statistically meaningful results. Furthermore, each algorithm was implemented using MATLAB 2013 and was run on an Intel Core i5 machine, 2.2 GHz CPU and 4 GB of RAM.

Table 2: Details of datasets

Dataset	No. of Attributes	No. of Objects	No. of Classes
Breastcancer	9	699	2
BreastEW	30	569	2
Exactly	13	1000	2
Exactly2	13	1000	2
HeartEW	13	270	2
Lymphography	18	148	4
M-of-n	13	1000	2
PenglungEW	325	73	7
SonarEW	60	208	2
SpectEW	22	267	2
CongressEW	16	435	2
IonosphereEW	34	351	2
KrvskpEW	36	3196	2
Tic-tac-toe	9	958	2
Vote	16	300	2
WaveformEW	40	5000	3
WineEW	13	178	3
Zoo	16	101	7

6.2. Parameter tuning

BDA is a metaheuristic algorithm that is highly dependent on the parameter values when searching for the optimal solutions. Setting the suitable parameter values requires an extra effort since it depends on the problem being solved. To determine the most suitable values of the proposed approaches, the following empirical studies were conducted.

16

The parameters that were included in these studies are population size, max number of iterations, α and β (in the fitness function), the number of nearest neighbors (k) in the k-NN classifier. In all initial empirical studies, the Leukemia dataset was used because it revealed more sensitivity compared to other datasets.

6.3. Assessment of the impact of population size and number of iterations

Table 3 compares the performance of the optimizer as it is monitored for slight adjustments in the parameter values. As it can be interpreted from Table 3, for a population size of 10 and 100 iterations, the optimizer can obtain the best results in terms of classification accuracy, and is very competitive in terms of selection ratio and time.

Table 3: Average accuracy, selection ration and time results, when using different combinations of population sizes and number of iterations for the Leukemia dataset.

Pop Size	Max Iterations	Accuracy	Selection Ration	Time
10	100	**1**	0.485	**212.11**
20	100	0.938	0.481	455.18
30	100	**1**	0.474	698.32
50	100	**1**	0.473	1262.41
10	150	**1**	0.477	318.44
20	150	0.933	0.477	685.21
30	150	0.933	**0.472**	697.86
50	150	0.916	0.481	1891.88

6.4. Assessment of the impact of α and β on the fitness function

FS problems cover two contradictory objectives (i.e., classification accuracy (maximization) and selection ration (minimization)) in the optimization process. In the fitness function (see Eq. 17), those two objectives are presented, and two values (α and β) were used to represent their weights for the user. That is, α determines the weight of classification accuracy, while β corresponds to the weight of the features reduction rate. In this section, an initial empirical study is presented to assess the influence of both α and β on the performance of the BDA optimizer. Different values for α and β were used to measure the classification accuracy and selection ration. The leukemia dataset was used in all experiments in this section due to its sensitivity in comparison with other datasets. Table 4 shows the classification accuracy and selection ration with different combinations of α and β values. Inspecting the results in Table 4, it can be seen that accuracy rate and the selection ration are changing with the different values of α and β. That is to say, when increasing alpha, the classification accuracy increases and the selection ratio decreases.

To make fair comparisons with the results obtained in previous works, we set $\alpha = 0.99$ and $\beta = 0.01$ which are commonly used values in the literature [55].

In the experiments, the K-NN classifier (with $K = 5$ [59]), with the Euclidean distance metric, was used to evaluate the feature subsets.

The parameters setting of the algorithms are outlined in Table 5.

17

Table 4: Impact of α and β on the accuracy and selection ration results for the Leukemia dataset.

Alpha	Beta	Accuracy	Selection Ration	Time
0.5	0.5	0.956	0.476	212.21
0.7	0.3	0.967	0.474	212.30
0.9	0.1	0.936	0.477	212.15
0.99	0.01	**1**	**0.471**	**211.91**

Table 5: The parameter settings

Parameter	Value
Population size	10
Number of iterations	100
Dimension	Number of features
Number of runs for each technique	30
α in fitness function	0.99
β in fitness function	0.01
a in GWO	[2 0]
Q_{min} Frequency minimum in BA	0
Q_{max} Frequency maximum in BA	2
A Loudness in BA	0.5
r Pulse rate in BA	0.5
G_0 in GSA	100
α in GSA	20

6.5. Results and discussion

To study the influence of the newly proposed time-dependent TFs on the performance of the BDA, this paper provides a comparison of the results obtained by BDA with eight static TFs and BDA, and those obtained by BDA with eight time-dependent TFs. The best reported approaches were then compared with state-of-the-art FS methods using two phases. In the first phase, we implemented three recent FS techniques (*i.e.* BGSA, BBA, and bGWO) and compared them with the proposed approaches. Then in the second phase we used some previously published results of some well-known algorithms (GA, PSO, and GWO). The comparisons are done using the following criteria:

- The mean classification accuracy was obtained from 30 runs. For each run, the accuracy of the best solution is considered.

- The average selection size from 30 runs. In each run, the cardinality of the best solution is considered.

- The average of the best fitness values obtained from each approach are reported.

- Statistical standard deviation (STD) is reported for all approaches to indicate the stability and robustness of the optimizer.

- Wilcoxon signed-rank test to assess the significance of the results obtained from the proposed approach.

18

- Specificity, Sensitivity, and Area Under the ROC Curve (AUC) [67] for binary datasets for each approach are reported.

Note that in the following tables, the best results are highlighted in bold.

The classification accuracy obtained from the approaches that are based on S-shaped TFs are reported in Table 6, followed by the results of the approaches based on the V-shaped TFs in Table 7. We named the approaches with the static TFs as S1 to S4 for the S-shaped TFs, and V1 to V4 for the V-shaped TFs. Similarly, the time dependent TFs are named as TV_{S1} to TV_{S4} for the S-shaped functions and TV_{V1} to TV_{V4} for the V-shaped TFs.

In general, it is observed that the BDA with time-dependent TFs ($TV - TFs$) perform better than those with fixed TFs. In Table 6, it is observed that TV_{S1} provided the best results on 50% of the datasets, and on three datasets it provided the same results as $S1$. In the case of TV_{S2}, the traditional TFs approach performs better than the TV approach, however, it can be found that TV_{S3} obtained the best results for eight datasets and is competitive with S3 for five datasets, while S3 obtained the best results for only five datasets. S4 and TV_{S4} obtained the same results. The results in Table 6 show that TV approaches are robust since they have the smallest standard deviation.

By observing the results of Table 7, it is obvious that TV performs well compared to traditional TFs based approaches. An exception is the case of V2 and TV_{V2} that obtained nearly the same results, while TV_{V1}, TV_{V3} and TV_{V4} obtained the best results for 70%, 72% and 77% of the datasets, respectively. The time TV-TFs with V-shaped functions highly improved the performance of the BDA. This is because a good balance is achieved between exploration and exploitation. The robustness of TV-BDA approaches is also observed as they obtained smaller values in terms of standard deviation.

In Tables 8 and 9, the ratio of kept feature to the total number of features using the BDA that utilize the traditional TFs and TV-BDA are reported. Once again, the TV-BDA outperforms TF-BDA approaches on the majority of the datasets. In particular, TV_{S1} selected the minimal number features for 10 datasets and was competitive with other approaches on the two other datasets.

The same observation can be made when studying the behaviors of TV_{S2}, TV_{S3}, and TV_{S4}, which achieved the best performance on 12, 13 and 11 datasets, respectively. Since the main objective of FS is to minimize the number of selected features without decreasing the classification accuracy as much as possible, we can say that S-shaped TV-BDA approaches showed good performance since they obtained comparative classification accuracy using a smaller number of features. The same observation can be made from Table 9, where TV-BDA approaches obtained comparative performance in terms of number of selected features, while the classification accuracy of those approaches is much better than TF-BDA approaches.

From Table 10, it is found that the S-shaped TV-BDA optimizers are better than TF-BDA in terms of fitness measures. TV_{S1} obtained the best fitness value on 50% of the datasets, and on two datasets it obtained the same results as $S1$. In the case of TV_{S3} and TV_{S4}, it can be detected that they obtained the best results for 67% and 55% of the datasets, respectively.

19

Table 6: Comparison between the original S-shaped transfer functions and time dependent variants in terms of accuracy

Benchmark	Measure	S1	TV_{S1}	S2	TV_{S2}	S3	TV_{S3}	S4	TV_{S4}
Breastcancer	AVG	0.9643	**0.9786**	0.9714	**0.9852**	**0.9929**	0.9929	**0.9786**	**0.9786**
	STD	0.0000	0.0000	0.0000	0.0018	0.0000	0.0000	0.0000	0.0000
BreastEW	AVG	**0.9839**	0.9716	**0.9889**	0.9769	**0.9822**	0.9792	0.9564	**0.9936**
	STD	0.0052	0.0072	0.0039	0.0075	0.0028	0.0049	0.0059	0.0056
Exactly	AVG	**1.0000**	**1.0000**	**1.0000**	0.9902	**1.0000**	**1.0000**	0.9998	**1.0000**
	STD	0.0000	0.0000	0.0000	0.0539	0.0000	0.0000	0.0009	0.0000
Exactly2	AVG	**0.7922**	0.7892	**0.7952**	0.7630	**0.7983**	0.7725	**0.7658**	0.7508
	STD	0.0141	0.0099	0.0100	0.0060	0.0040	0.0222	0.0116	0.0057
HeartEW	AVG	0.9154	**0.9377**	**0.8691**	0.8679	**0.8846**	0.8759	0.8759	**0.9080**
	STD	0.0173	0.0091	0.0047	0.0193	0.0080	0.0139	0.0170	0.0191
Lymphography	AVG	0.9440	**0.9667**	0.9510	**0.9608**	**0.9844**	0.9922	0.9822	**0.9889**
	STD	0.0218	0.0000	0.0168	0.0215	0.0169	0.0143	0.0169	0.0183
M-of-n	AVG	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
penglungEW	AVG	**1.0000**	0.9044	**1.0000**	**1.0000**	0.9289	**1.0000**	**0.9889**	0.9600
	STD	0.0000	0.0336	0.0000	0.0000	0.0169	0.0000	0.0253	0.0332
SonarEW	AVG	0.9825	**0.9992**	**1.0000**	**1.0000**	0.9683	**0.9841**	**0.9865**	0.9651
	STD	0.0165	0.0043	0.0000	0.0000	0.0144	0.0114	0.0135	0.0136
SpectEW	AVG	0.8685	**0.9352**	**0.9049**	0.8617	**0.9228**	0.8519	**0.8741**	0.8438
	STD	0.0141	0.0117	0.0064	0.0094	0.0120	0.0109	0.0123	0.0093
CongressEW	AVG	0.9820	**0.9977**	**0.9885**	0.9502	**0.9966**	0.9866	**0.9766**	0.9751
	STD	0.0065	0.0047	0.0000	0.0063	0.0061	0.0044	0.0021	0.0044
IonosphereEW	AVG	**0.9634**	0.9535	**0.9582**	0.9347	0.9728	**0.9911**	**0.9639**	0.9075
	STD	0.0102	0.0092	0.0094	0.0094	0.0051	0.0094	0.0096	0.0121
KrvskpEW	AVG	0.9853	**0.9858**	0.9842	**0.9912**	0.9787	**0.9794**	0.9754	**0.9840**
	STD	0.0030	0.0020	0.0031	0.0029	0.0048	0.0031	0.0040	0.0032
Tic-tac-toe	AVG	**0.8431**	0.8267	**0.8594**	0.8333	0.8323	**0.8469**	**0.8490**	0.8290
	STD	0.0018	0.0049	0.0000	0.0000	0.0021	0.0064	0.0000	0.0099
Vote	AVG	**0.9739**	0.9694	0.9828	**0.9850**	0.9667	**0.9894**	0.9783	**0.9994**
	STD	0.0084	0.0077	0.0030	0.0067	0.0000	0.0082	0.0078	0.0030
WaveformEW	AVG	0.7544	**0.7630**	0.7581	**0.7733**	0.7542	**0.7580**	0.7534	**0.7621**
	STD	0.0078	0.0064	0.0074	0.0086	0.0078	0.0066	0.0058	0.0077
WineEW	AVG	**1.0000**	0.9750	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	0.0000	0.0085	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Zoo	AVG	**1.0000**	**1.0000**	0.9524	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Ranking	W|T|L	7|3|8	8|3|7	8|4|6	6|4|8	5|5|8	8|5|5	7|4|7	7|4|7

20

Table 7: Comparison between the original V-shaped transfer functions and time dependent variants in terms of accuracy results

Benchmark	Measure	V1	TV_{V1}	V2	TV_{V2}	V3	TV_{V3}	V4	TV_{V4}
Breastcancer	AVG	**0.9786**	0.9671	**0.9805**	0.9781	0.9569	**0.9895**	0.9619	**0.9771**
	STD	0.0050	0.0036	0.0032	0.0072	0.0083	0.0070	0.0057	0.0047
BreastEW	AVG	0.9465	**0.9582**	0.9816	0.9658	0.9520	**0.9801**	0.9494	**0.9740**
	STD	0.0146	0.0121	0.0070	0.0093	0.0091	0.0065	0.0119	0.0096
Exactly	AVG	**0.8847**	0.8605	**0.9075**	0.8947	0.9093	**0.9137**	0.8838	**0.9290**
	STD	0.0860	0.0971	0.0891	0.0814	0.0873	0.0841	0.1051	0.0551
Exactly2	AVG	0.7230	**0.7428**	0.7190	**0.7595**	0.6985	**0.7358**	**0.7417**	0.7257
	STD	0.0216	0.0214	0.0195	0.0166	0.0209	0.0216	0.0167	0.0200
HeartEW	AVG	0.8463	**0.8617**	0.8525	0.8525	0.7926	**0.8321**	0.8148	**0.8858**
	STD	0.0234	0.0251	0.0269	0.0250	0.0235	0.0153	0.0188	0.0327
Lymphography	AVG	0.8877	**0.8967**	0.8673	0.7956	0.8744	**0.9121**	0.8611	**0.8950**
	STD	0.0296	0.0308	0.0309	0.0287	0.0243	0.0268	0.0264	0.0321
M-of-n	AVG	0.9493	**0.9523**	0.9518	0.9450	**0.9673**	0.9578	0.9463	**0.9732**
	STD	0.0383	0.0473	0.0373	0.0435	0.0301	0.0405	0.0557	0.0275
penglungEW	AVG	0.8733	**0.9333**	**0.9778**	0.9556	**1.0000**	0.9222	**0.9333**	0.8067
	STD	0.0203	0.0000	0.0320	0.0320	0.0000	0.0466	0.0000	0.0441
SonarEW	AVG	**0.9476**	0.9214	**0.9595**	0.9587	**0.9659**	0.9563	0.9722	**0.9952**
	STD	0.0229	0.0251	0.0243	0.0216	0.0162	0.0251	0.0199	0.0097
SpectEW	AVG	0.8580	**0.8599**	**0.8525**	0.8099	0.7796	**0.8198**	**0.8969**	0.8765
	STD	0.0235	0.0232	0.0165	0.0279	0.0171	0.0168	0.0256	0.0164
CongressEW	AVG	0.9521	**0.9586**	0.9548	**0.9713**	0.9678	**0.9705**	0.9686	**0.9950**
	STD	0.0091	0.0119	0.0113	0.0108	0.0093	0.0130	0.0074	0.0072
IonosphereEW	AVG	**0.9535**	0.9437	0.9582	**0.9779**	0.9042	**0.9700**	0.9131	**0.9249**
	STD	0.0182	0.0133	0.0120	0.0096	0.0186	0.0121	0.0129	0.0130
KrvskpEW	AVG	0.9620	**0.9664**	0.9666	0.9581	0.9649	**0.9733**	0.9658	**0.9705**
	STD	0.0100	0.0105	0.0082	0.0120	0.0119	0.0079	0.0163	0.0088
Tic-tac-toe	AVG	0.8153	**0.8227**	0.7939	**0.8021**	0.8347	0.8017	0.7915	**0.8215**
	STD	0.0320	0.0238	0.0171	0.0269	0.0208	0.0225	0.0209	0.0196
Vote	AVG	0.9489	**0.9544**	0.9361	**0.9733**	0.9594	**0.9683**	0.9733	0.9617
	STD	0.0131	0.0175	0.0146	0.0155	0.0129	0.0119	0.0136	0.0089
WaveformEW	AVG	0.7382	**0.7456**	0.7493	**0.7554**	0.7404	**0.7496**	0.7433	**0.7488**
	STD	0.0111	0.0098	0.0105	0.0113	0.0089	0.0100	0.0125	0.0097
WineEW	AVG	**0.9769**	0.9537	0.9778	**0.9778**	0.9981	**1.0000**	0.9991	0.9991
	STD	0.0105	0.0234	0.0153	0.0170	0.0070	0.0000	0.0051	0.0051
Zoo	AVG	**1.0000**	**1.0000**	0.9555	**0.9984**	0.9921	0.9825	0.9524	**0.9825**
	STD	0.0000	0.0000	0.0121	0.0087	0.0181	0.0233	0.0354	0.0233
Ranking	W|T|L	5|1|12	12|1|5	10|0|8	8|0|10	5|0|13	13|0|5	4|1|13	13|1|4

Table 8: Comparison between the original S-shaped transfer functions and time dependent variants in terms of minimum number of selected features

Benchmark	Measure	S1	TV_{S1}	S2	TV_{S2}	S3	TV_{S3}	S4	TV_{S4}
Breastcancer	AVG	**6.00**	7.00	5.00	**4.93**	**5.00**	**5.00**	**4.00**	**4.00**
	STD	0.00	0.00	0.00	0.25	0.00	0.00	0.00	0.00
BreastEW	AVG	12.83	**12.63**	13.43	**11.47**	13.07	**11.50**	**13.00**	13.33
	STD	1.97	2.06	1.50	2.05	2.15	2.15	2.23	2.11
Exactly	AVG	**6.00**	**6.00**	6.03	**5.97**	6.13	**6.00**	6.33	**6.00**
	STD	0.00	0.00	0.18	0.18	0.35	0.00	0.48	0.00
Exactly2	AVG	**8.23**	8.33	**6.33**	8.93	**2.40**	7.10	**5.03**	8.90
	STD	3.07	2.96	2.68	0.58	1.52	2.72	4.11	1.42
HeartEW	AVG	**4.97**	7.30	6.60	**5.93**	6.63	**5.77**	**6.40**	6.63
	STD	0.18	1.12	0.86	1.34	0.93	1.50	1.10	1.67
Lymphography	AVG	7.97	**5.90**	8.20	**7.80**	8.73	**8.23**	**6.53**	7.17
	STD	2.14	1.49	1.52	2.07	1.05	2.73	2.27	1.76
M-of-n	AVG	**6.00**	**6.00**	**6.00**	**6.00**	6.13	**6.03**	6.20	**6.00**
	STD	0.00	0.00	0.00	0.00	0.35	0.18	0.41	0.00
penglungEW	AVG	125.80	**121.40**	129.87	**116.97**	145.13	**121.17**	144.73	**136.33**
	STD	4.58	8.29	5.16	3.76	6.97	4.71	8.69	9.35
SonarEW	AVG	31.23	**27.73**	26.73	**22.53**	28.00	**25.57**	27.87	**25.40**
	STD	6.06	3.11	2.69	2.60	3.44	3.35	4.00	3.64
SpectEW	AVG	**8.07**	8.47	**9.17**	9.57	8.80	**6.83**	9.53	**7.40**
	STD	2.26	1.57	1.18	2.28	1.86	2.56	1.72	2.65
CongressEW	AVG	6.13	**4.50**	**2.17**	6.87	**5.43**	5.53	6.53	**5.70**
	STD	1.48	1.14	0.38	1.14	0.97	2.29	1.11	1.21
IonosphereEW	AVG	13.53	**12.43**	13.77	**12.60**	14.37	**11.47**	13.93	**12.17**
	STD	2.70	2.43	2.54	2.27	2.82	1.89	2.41	2.91
KrvskpEW	AVG	22.50	**19.67**	21.37	**21.17**	22.57	**20.67**	21.53	**20.67**
	STD	2.76	2.59	2.71	1.97	2.73	2.32	2.47	1.86
Tic-tac-toe	AVG	**6.60**	6.87	7.00	**6.00**	**6.20**	7.00	7.00	**6.67**
	STD	1.04	0.35	0.00	0.00	0.41	0.00	0.00	0.76
Vote	AVG	**4.13**	5.30	6.30	**5.83**	4.73	**3.37**	5.57	5.83
	STD	0.82	1.70	1.15	1.78	1.82	1.43	2.73	1.78
WaveformEW	AVG	26.07	**25.03**	**23.97**	25.53	23.53	**23.00**	23.37	**21.33**
	STD	3.49	4.58	3.74	2.96	3.14	3.18	2.97	2.88
WineEW	AVG	5.07	**4.50**	**3.40**	4.33	4.57	**3.57**	4.87	**3.93**
	STD	0.25	0.82	0.86	0.48	0.50	0.63	0.63	0.64
Zoo	AVG	4.87	**2.70**	6.17	**3.77**	3.43	4.43	**1.73**	3.17
	STD	0.63	0.60	0.38	0.73	0.50	0.50	0.58	0.38
Ranking	W|T|L	6|2|10	10|2|6	5|1|12	12|1|5	4|1|13	13|1|4	6|1|11	11|1|6

22

Table 9: Comparison between the original V-shaped transfer functions and time dependent variants in terms of minimum number of selected features

Benchmark	Measure	V1	TV_{V1}	V2	TV_{V2}	V3	TV_{V3}	V4	TV_{V4}
Breastcancer	AVG	**5.23**	5.63	**4.60**	5.23	**4.77**	5.27	5.47	**5.37**
	STD	1.30	1.30	0.93	0.77	0.97	0.74	1.43	0.81
BreastEW	AVG	**16.50**	18.17	**17.87**	18.43	**16.17**	17.50	16.77	**16.37**
	STD	2.89	3.03	2.64	3.57	2.64	2.56	2.19	2.47
Exactly	AVG	**9.10**	9.33	**8.90**	9.43	8.90	**8.87**	**8.60**	8.97
	STD	1.32	1.30	2.09	1.45	1.30	1.14	1.16	0.93
Exactly2	AVG	8.87	**6.90**	8.87	**5.53**	**7.37**	9.27	**8.57**	9.67
	STD	1.43	3.01	2.70	2.57	2.59	2.29	1.70	1.37
HeartEW	AVG	**7.53**	7.73	8.70	**7.87**	**7.47**	9.73	**7.87**	9.07
	STD	2.10	1.68	1.84	1.68	1.83	1.48	1.50	1.53
Lymphography	AVG	**11.77**	13.00	**10.30**	11.20	11.33	**10.47**	11.03	**10.80**
	STD	2.56	2.13	2.72	2.20	1.75	1.72	2.24	1.73
M-of-n	AVG	9.57	**8.93**	9.37	**9.20**	**8.60**	8.63	8.50	**8.43**
	STD	1.65	1.46	1.16	1.10	0.77	1.10	1.41	1.14
penglungEW	AVG	**151.90**	155.50	171.83	**161.57**	**156.03**	174.20	**152.17**	165.40
	STD	9.90	8.01	24.62	15.75	7.81	19.05	6.12	15.90
SonarEW	AVG	**35.53**	37.07	35.47	**34.37**	**33.63**	35.97	**33.07**	35.33
	STD	4.59	4.62	4.15	5.18	3.97	4.19	2.89	4.88
SpectEW	AVG	12.53	**12.33**	12.63	**11.73**	**10.63**	13.60	12.77	**11.20**
	STD	2.52	2.45	2.30	2.33	2.98	2.99	2.13	1.67
CongressEW	AVG	**8.63**	9.40	9.17	**8.87**	9.30	**8.60**	9.07	**7.10**
	STD	1.40	2.62	1.72	1.61	1.62	2.11	1.78	1.95
IonosphereEW	AVG	**18.73**	19.40	20.67	**18.57**	18.23	18.37	18.13	**17.53**
	STD	3.90	2.65	3.09	3.54	3.07	2.93	2.27	3.18
KrvskpEW	AVG	27.10	**26.93**	**26.53**	26.63	26.40	**26.30**	25.20	**25.13**
	STD	2.25	2.05	2.49	3.36	2.16	2.81	3.07	2.60
Tic-tac-toe	AVG	7.33	**6.70**	**6.87**	7.10	7.30	**6.40**	**6.30**	6.53
	STD	1.30	1.24	0.90	1.16	0.92	1.07	0.70	0.57
Vote	AVG	9.80	**8.37**	**8.63**	10.23	7.83	**7.10**	**7.40**	8.47
	STD	2.04	1.63	1.88	2.27	1.46	2.04	1.81	2.01
WaveformEW	AVG	**30.43**	30.90	30.57	**29.37**	**28.80**	29.07	**27.03**	30.57
	STD	3.88	4.00	3.11	3.05	3.94	3.32	4.42	3.86
WineEW	AVG	8.20	**7.37**	**7.00**	7.57	7.50	**7.40**	7.10	**6.67**
	STD	1.69	1.52	0.87	1.63	1.17	1.45	1.77	1.37
Zoo	AVG	7.17	**5.60**	**7.57**	8.67	9.20	**9.03**	**8.43**	8.47
	STD	1.32	1.30	1.33	1.52	1.99	1.90	1.57	1.36
Ranking	W|T|L	10|0|8	8|0|10	9|0|9	9|0|9	9|0|9	9|0|9	8|0|10	10|0|8

23

Given that TV-BDA approaches have the lowest standard deviation as presented in Table 11, TV-based approaches can be considered as more robust than approaches based on traditional TFs. According to results of Table 11, V-shaped TV-BDA approaches outperform traditional V-shaped approaches for most datasets, where TV_{V1} outperformed V1 on 72% of the datasets, and TV_{V1}, and TV_{V1} are better for 72% and 77% of the datasets, respectively.

These results indicate how important is the role of the TF in BDA, since by selecting a suitable function, the performance of the BDA can be remarkably increased. Moreover, it is clear that adapting the behavior of the TF through the optimization process has a major influence on improving the performance of the BDA. Furthermore, the good performance of the BDA highlights its ability at searching the feature space for the most informative features and avoid premature convergence that may be caused by falling in local optima.

In addition, the improved potential to balance between exploration and exploitation throughout iterations is another reason for the BDA's superiority. Since the only modification is the replacement of the traditional TFs with the TV-TFs, these results show that the adaptive control of the TFs can significantly improve the search ability of the BDA. Overall, we can say that TV_{S3} is the best approach when compared with S-shaped approaches, while TV_{V4}) is the best among the V-shaped approaches. In the next subsection, we assess the performance of those two approaches by comparing their performances with other well-known state-of-the-art FS algorithms.

6.6. Comparison with other metaheuristic based approaches

After analyzing the results of the proposed approaches, this section presents a comparison between the best two approaches proposed in this work (TV_{S3} and TV_{V4}) and the most popular metaheuristics-based feature selection algorithms (BGWO, BGSA, BPSO, BBA, and GA).

The performance of TV_{S3} and TV_{V4} is compared in terms of average accuracy and standard deviation. As per results in Table 12, TV_{S3} performs better than TV_{V4}. It obtained the best results for 78% of the datasets. Moreover, we can observe that the performance of the proposed TV_{S3} approach is better than all the other algorithms for all datasets in terms of classification accuracy. Figure 5 compares TV_{S3}, TV_{V4} and other approaches in terms of classification accuracy. Based on reduction rates presented in Table 13, it is observed that TV_{S3} outperforms other algorithms for 10 out of 18 datasets, while the BBA and the BGSA obtained the best results for seven datasets and one dataset, respectively. The obtained reduction rates are represented in Fig. 6. Table 14 shows the superiority of TV_{S3}, where the reported fitness combines both classification accuracy and reduction rate, and showed that TV_{S3} had the best values for 14 out of 18 datasets.

Table 10: Comparison between the original transfer S-shaped functions and time dependent S-shaped variants in terms of best fitness values

Benchmark	Measure	S1	TV_{S1}	S2	TV_{S2}	S3	TV_{S3}	S4	TV_{S4}
Breastcancer	AVG	0.0420	**0.0290**	0.0338	**0.0201**	0.0126	0.0126	**0.0257**	**0.0257**
	STD	0.0000	0.0000	0.0000	0.0015	0.0000	0.0000	0.0000	0.0000
BreastEW	AVG	**0.0202**	0.0323	**0.0155**	0.0267	**0.0220**	0.0244	0.0475	**0.0108**
	STD	0.0049	0.0070	0.0038	0.0074	0.0028	0.0048	0.0056	0.0053
Exactly	AVG	**0.0046**	**0.0046**	**0.0046**	0.0143	0.0047	**0.0046**	0.0050	**0.0046**
	STD	0.0000	0.0000	0.0001	0.0532	0.0003	0.0000	0.0011	0.0000
Exactly2	AVG	**0.2121**	0.2151	**0.2077**	0.2415	**0.2015**	0.2307	**0.2357**	0.2535
	STD	0.0141	0.0099	0.0081	0.0058	0.0030	0.0202	0.0084	0.0048
HeartEW	AVG	0.0875	**0.0673**	0.1346	0.1353	**0.1194**	0.1273	0.1278	**0.0962**
	STD	0.0171	0.0083	0.0052	0.0187	0.0078	0.0143	0.0166	0.0177
Lymphography	AVG	0.0598	**0.0363**	0.0530	**0.0431**	0.0203	**0.0123**	0.0212	**0.0150**
	STD	0.0214	0.0008	0.0162	0.0210	0.0165	0.0133	0.0165	0.0180
M-of-n	AVG	**0.0046**	0.0046	**0.0046**	**0.0046**	0.0047	**0.0046**	0.0048	**0.0046**
	STD	0.0000	0.0000	0.0000	0.0000	0.0003	0.0001	0.0003	0.0000
penglungEW	AVG	**0.0039**	0.0983	0.0040	**0.0036**	0.0749	**0.0037**	**0.0155**	0.0438
	STD	0.0001	0.0331	0.0002	0.0001	0.0166	0.0001	0.0248	0.0327
SonarEW	AVG	0.0225	**0.0054**	0.0045	**0.0038**	0.0361	**0.0200**	0.0180	0.0388
	STD	0.0157	0.0042	0.0004	0.0004	0.0141	0.0111	0.0131	0.0133
SpectEW	AVG	0.1338	**0.0680**	**0.0983**	0.1412	**0.0804**	0.1498	**0.1290**	0.1580
	STD	0.0134	0.0118	0.0065	0.0086	0.0116	0.0100	0.0118	0.0085
CongressEW	AVG	0.0217	**0.0051**	0.0127	0.0536	**0.0068**	0.0167	0.0272	0.0282
	STD	0.0064	0.0041	0.0002	0.0058	0.0059	0.0040	0.0017	0.0044
IonosphereEW	AVG	**0.0402**	0.0497	0.0454	0.0683	0.0312	**0.0122**	0.0399	0.0951
	STD	0.0102	0.0093	0.0093	0.0093	0.0049	0.0095	0.0093	0.0120
KrvskpEW	AVG	0.0208	**0.0195**	0.0216	**0.0146**	0.0274	**0.0262**	0.0304	**0.0216**
	STD	0.0028	0.0022	0.0031	0.0030	0.0046	0.0033	0.0039	0.0031
Tic-tac-toe	AVG	**0.1627**	0.1792	**0.1470**	0.1717	0.1729	**0.1594**	**0.1573**	0.1767
	STD	0.0006	0.0046	0.0000	0.0000	0.0025	0.0063	0.0000	0.0089
Vote	AVG	**0.0284**	0.0336	0.0210	**0.0185**	0.0360	**0.0126**	0.0249	**0.0042**
	STD	0.0083	0.0070	0.0027	0.0066	0.0011	0.0073	0.0069	0.0031
WaveformEW	AVG	0.2496	**0.2409**	0.2455	**0.2308**	0.2492	**0.2453**	0.2500	**0.2408**
	STD	0.0077	0.0069	0.0074	0.0084	0.0075	0.0070	0.0059	0.0077
WineEW	AVG	**0.0039**	0.0282	**0.0026**	0.0033	0.0035	**0.0027**	0.0037	**0.0030**
	STD	0.0002	0.0083	0.0007	0.0004	0.0004	0.0005	0.0005	0.0005
Zoo	AVG	0.0030	**0.0017**	0.0510	**0.0024**	**0.0021**	0.0028	0.0011	**0.0020**
	STD	0.0004	0.0004	0.0002	0.0005	0.0003	0.0003	0.0004	0.0002
Ranking	W|T|L	7|2|9	**9|2|7**	9|1|8	8|1|9	6|1|11	**11|1|6**	8|1|9	**9|1|8**

25

Table 11: Comparison between the original transfer V-shaped functions and time-dependent V-shaped variants in terms of best fitness values

| Benchmark	Measure	V1	TV_{V1}	V2	TV_{V2}	V3	TV_{V3}	V4	TV_{V4}
Breastcancer	AVG	**0.0270**	0.0388	**0.0244**	0.0275	0.0480	**0.0162**	0.0438	**0.0286**
	STD	0.0052	0.0040	0.0031	0.0067	0.0081	0.0072	0.0060	0.0048
BreastEW	AVG	0.0585	**0.0475**	0.0242	0.0400	0.0529	**0.0255**	0.0557	**0.0312**
	STD	0.0147	0.0120	0.0071	0.0090	0.0091	0.0063	0.0117	0.0092
Exactly	AVG	**0.1212**	0.1453	**0.0984**	0.1115	0.0966	**0.0923**	0.1216	**0.0772**
	STD	0.0861	0.0966	0.0886	0.0814	0.0868	0.0840	0.1045	0.0553
Exactly2	AVG	0.2810	**0.2599**	0.2850	**0.2424**	0.3042	**0.2687**	0.2623	0.2790
	STD	0.0213	0.0208	0.0190	0.0173	0.0202	0.0209	0.0165	0.0195
HeartEW	AVG	0.1580	**0.1428**	0.1527	**0.1521**	0.2111	**0.1737**	0.1894	**0.1200**
	STD	0.0237	0.0247	0.0260	0.0251	0.0227	0.0147	0.0188	0.0315
Lymphography	AVG	0.1178	**0.1095**	0.1371	0.2086	0.1306	**0.0928**	0.1436	**0.1099**
	STD	0.0290	0.0303	0.0303	0.0287	0.0238	0.0268	0.0260	0.0314
M-of-n	AVG	0.0575	**0.0541**	0.0549	0.0615	**0.0390**	0.0484	0.0597	**0.0331**
	STD	0.0390	0.0479	0.0377	0.0438	0.0302	0.0408	0.0557	0.0280
penglungEW	AVG	0.1301	**0.0708**	**0.0273**	0.0490	**0.0048**	0.0824	**0.0707**	0.1965
	STD	0.0199	0.0002	0.0314	0.0313	0.0002	0.0460	0.0002	0.0435
SonarEW	AVG	**0.0578**	0.0840	**0.0460**	0.0466	**0.0394**	0.0492	0.0330	**0.0106**
	STD	0.0228	0.0248	0.0242	0.0213	0.0157	0.0245	0.0198	0.0096
SpectEW	AVG	0.1463	**0.1443**	**0.1518**	0.1936	0.2230	**0.1846**	0.1079	0.1273
	STD	0.0229	0.0233	0.0167	0.0282	0.0171	0.0167	0.0254	0.0162
CongressEW	AVG	0.0528	**0.0468**	0.0505	**0.0340**	0.0377	**0.0346**	0.0368	**0.0094**
	STD	0.0092	0.0113	0.0111	0.0102	0.0089	0.0131	0.0074	0.0078
IonosphereEW	AVG	**0.0515**	0.0615	0.0474	**0.0273**	0.1002	**0.0351**	0.0913	**0.0795**
	STD	0.0183	0.0133	0.0120	0.0097	0.0185	0.0122	0.0130	0.0131
KrvskpEW	AVG	0.0451	**0.0407**	0.0404	0.0489	0.0421	**0.0338**	0.0408	**0.0362**
	STD	0.0101	0.0105	0.0082	0.0120	0.0119	0.0078	0.0161	0.0089
Tic-tac-toe	AVG	0.1910	**0.1829**	0.2116	**0.2038**	**0.1717**	0.2034	0.2134	**0.1839**
	STD	0.0316	0.0237	0.0172	0.0273	0.0211	0.0227	0.0212	0.0194
Vote	AVG	0.0567	**0.0503**	0.0686	**0.0328**	0.0450	**0.0358**	0.0310	0.0432
	STD	0.0133	0.0177	0.0142	0.0150	0.0130	0.0119	0.0138	0.0086
WaveformEW	AVG	0.2668	**0.2596**	0.2558	**0.2495**	0.2642	**0.2552**	0.2609	**0.2563**
	STD	0.0108	0.0097	0.0107	0.0112	0.0089	0.0100	0.0125	0.0096
WineEW	AVG	**0.0292**	0.0515	**0.0274**	0.0278	0.0076	**0.0057**	0.0064	**0.0060**
	STD	0.0111	0.0229	0.0150	0.0161	0.0067	0.0011	0.0046	0.0052
Zoo	AVG	0.0045	**0.0035**	0.0488	**0.0070**	0.0136	0.0229	0.0524	**0.0226**
	STD	0.0008	0.0008	0.0121	0.0087	0.0172	0.0222	0.0346	0.0224
Ranking	W|T|L	5|0|13	**13|0|5**	10|0|8	8|0|10	5|0|13	**13|0|5**	4|0|14	**14|0|4**

26
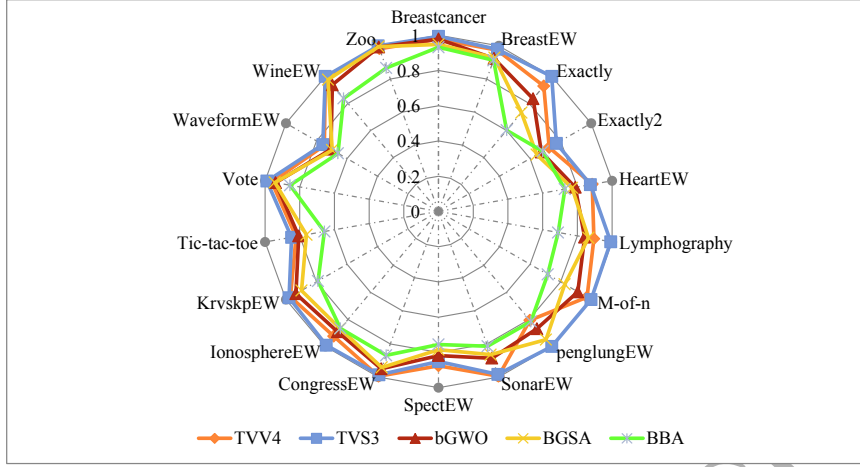
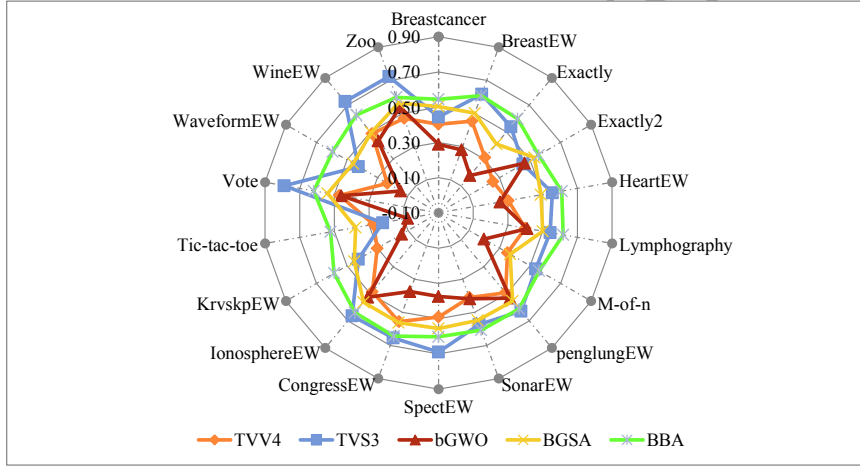Figure 5: Comparison of TV_{V4} and TV_{S3} versus other optimizers based on accuracy metric

Figure 6: Comparison between TVV4, TVS3 and other meta-heuristics in terms of feature reduction rate

Table 15 reports the results of the best BDA V-shaped and BDA S-shaped time-dependent variants versus others algorithms for the specificity metric. Results of Table 15 show that the proposed BDA-based techniques can outperform all other methods for that metric. For 69.23% of the datasets, the TV_{V4} variant performs best, while for the rest of the cases; TV_{S3} has better results.

Table 16 compares the best BDA V-shaped and BDA S-shaped time-dependent variants and others optimizers for the sensitivity metric. From Table 16, we see that the proposed BDA-based techniques can outperform other competitors according to the sensitivity measure. For 61.53% of the datasets, the TV_{V4} variant demonstrates satisfactory performance. But TV_{S3} has better performance for the rest of the cases except for SpectEW, where the BGSA has the highest sensitivity.

Table 17 compares results of the best BDA V-shaped and BDA S-shaped time-varying variants with others optimizers for the AUC metric. It is observed that the proposed

Table 12: Comparison between the best BDA V-shaped and BDA S-shaped time dependent variants and other meta-heuristics in term of accuracy

Benchmark	Measure	BDA		bGWO	BGSA	BBA
		TV_{V4}	TV_{S3}			
Breastcancer	AVG	0.9771	**0.9929**	0.9779	0.9481	0.9321
	STD	0.0047	0.0000	0.0103	0.0203	0.0513
BreastEW	AVG	0.9740	**0.9792**	0.9231	0.9284	0.9129
	STD	0.0096	0.0049	0.0152	0.0140	0.0349
Exactly	AVG	0.9290	**1.0000**	0.8345	0.7323	0.6015
	STD	0.0551	0.0000	0.0773	0.1244	0.0555
Exactly2	AVG	0.7257	**0.7725**	0.6740	0.6438	0.6830
	STD	0.0200	0.0222	0.0405	0.0407	0.0400
HeartEW	AVG	**0.8858**	0.8759	0.7883	0.7698	0.7284
	STD	0.0327	0.0139	0.0391	0.0664	0.0606
Lymphography	AVG	0.8950	**0.9922**	0.8422	0.8642	0.6894
	STD	0.0321	0.0143	0.0567	0.0805	0.1033
M-of-n	AVG	0.9732	**1.0000**	0.9130	0.8268	0.7155
	STD	0.0275	0.0000	0.0517	0.0608	0.0832
penglungEW	AVG	0.8067	**1.0000**	0.8689	0.9493	0.8156
	STD	0.0441	0.0000	0.0122	0.0543	0.0545
SonarEW	AVG	**0.9952**	0.9841	0.8865	0.8651	0.8143
	STD	0.0097	0.0114	0.0404	0.0465	0.0588
SpectEW	AVG	**0.8765**	0.8519	0.8179	0.7846	0.7556
	STD	0.0164	0.0109	0.0288	0.0342	0.0393
CongressEW	AVG	**0.9950**	0.9866	0.9502	0.9425	0.8686
	STD	0.0072	0.0044	0.0469	0.0260	0.0803
IonosphereEW	AVG	0.9249	**0.9911**	0.8911	0.8685	0.8662
	STD	0.0130	0.0094	0.0251	0.0257	0.0271
KrvskpEW	AVG	0.9705	**0.9794**	0.9346	0.8978	0.7898
	STD	0.0088	0.0031	0.0190	0.0526	0.0896
Tic-tac-toe	AVG	0.8215	**0.8469**	0.8057	0.7611	0.6578
	STD	0.0196	0.0064	0.0287	0.0380	0.0805
Vote	AVG	0.9617	**0.9894**	0.9394	0.9433	0.8556
	STD	0.0089	0.0082	0.0208	0.0246	0.1016
WaveformEW	AVG	0.7488	**0.7580**	0.7050	0.6971	0.6592
	STD	0.0097	0.0066	0.0154	0.0205	0.0460
WineEW	AVG	0.9991	**1.0000**	0.9380	0.9759	0.8380
	STD	0.0051	0.0000	0.0362	0.0348	0.1305
Zoo	AVG	0.9825	**1.0000**	0.9930	0.9952	0.8667
	STD	0.0233	0.0000	0.0229	0.0148	0.1142
Ranking	W|T|L	4|0|14	14|0|4	0|0|18	0|0|18	0|0|18

Table 13: Comparison between the best BDA V-shaped and BDA S-shaped time dependent variants and other meta-heuristics in terms of minimum number of features

Benchmark	Measure	BDA		bGWO	BGSA	BBA
		TV_{V4}	TV_{S3}			
Breastcancer	AVG	5.37	5.00	6.40	4.47	**4.10**
	STD	0.81	0.00	1.75	1.01	1.27
BreastEW	AVG	16.37	**11.50**	21.57	14.93	11.77
	STD	2.47	2.15	4.80	2.00	3.94
Exactly	AVG	8.97	6.00	10.70	7.67	**5.23**
	STD	0.93	0.00	2.02	1.49	2.25
Exactly2	AVG	9.67	7.10	6.97	6.13	**5.77**
	STD	1.37	2.72	2.74	2.08	1.57
HeartEW	AVG	9.07	5.77	9.70	6.63	**5.07**
	STD	1.53	1.50	1.99	1.94	1.70
Lymphography	AVG	10.80	8.23	10.60	9.00	**6.87**
	STD	1.73	2.73	2.63	2.18	1.96
M-of-n	AVG	8.43	6.03	10.43	8.20	**5.73**
	STD	1.14	0.18	1.45	1.16	1.82
penglungEW	AVG	165.40	**121.17**	152.33	145.10	126.47
	STD	15.90	4.71	7.00	4.88	15.62
SonarEW	AVG	35.33	25.57	34.87	27.07	**23.53**
	STD	4.88	3.35	7.81	3.64	5.15
SpectEW	AVG	11.20	**6.83**	13.77	9.77	8.73
	STD	1.67	2.56	2.93	2.30	2.29
CongressEW	AVG	7.10	**5.53**	10.00	7.00	5.70
	STD	1.95	2.29	1.88	1.91	2.18
IonosphereEW	AVG	17.53	**11.47**	16.17	14.90	12.30
	STD	3.18	1.89	2.35	2.89	3.40
KrvskpEW	AVG	25.13	20.67	30.90	19.73	**14.97**
	STD	2.60	2.32	2.93	2.36	2.88
Tic-tac-toe	AVG	6.53	7.00	8.30	5.60	**4.30**
	STD	0.57	0.00	1.24	0.97	1.70
Vote	AVG	8.47	**3.37**	8.63	7.37	6.10
	STD	2.01	1.43	2.63	1.67	2.14
WaveformEW	AVG	30.57	23.00	34.07	21.60	**16.23**
	STD	3.86	3.18	4.48	3.69	4.08
WineEW	AVG	6.67	**3.57**	7.37	6.57	4.87
	STD	1.37	0.63	1.67	1.36	1.87
Zoo	AVG	8.47	**4.43**	7.37	6.97	6.43
	STD	1.36	0.50	1.63	1.25	1.83
Ranking	W|T|L	0|0|18	8|0|10	0|0|18	0|0|18	10|0|8

Table 14: Comparison between the best BDA V-shaped and BDA S-shaped time dependent variants and other meta-heuristics in terms of fitness results

Benchmark	Measure	BDA		bGWO	BGSA	BBA
		TV_{V4}	TV_{S3}			
Breastcancer	AVG	0.0286	**0.0126**	0.0155	0.0273	0.0361
	STD	0.0048	0.0000	0.0023	0.0068	0.0050
BreastEW	AVG	0.0312	**0.0244**	0.0428	0.0390	0.0358
	STD	0.0092	0.0048	0.0066	0.0100	0.0087
Exactly	AVG	0.0772	**0.0046**	0.1851	0.2532	0.3028
	STD	0.0553	0.0000	0.0515	0.0944	0.1080
Exactly2	AVG	0.2790	**0.2307**	0.2489	0.2876	0.2499
	STD	0.0195	0.0202	0.0136	0.0137	0.0151
HeartEW	AVG	**0.1200**	0.1273	0.1285	0.1369	0.1613
	STD	0.0315	0.0143	0.0258	0.0299	0.0230
Lymphography	AVG	0.1099	**0.0123**	0.0832	0.0813	0.1622
	STD	0.0314	0.0133	0.0346	0.0329	0.0526
M-of-n	AVG	0.0331	**0.0046**	0.0874	0.1652	0.1653
	STD	0.0280	0.0001	0.0387	0.0413	0.0438
penglungEW	AVG	0.1965	**0.0037**	0.1257	0.0045	0.1317
	STD	0.0435	0.0001	0.0249	0.0002	0.0384
SonarEW	AVG	**0.0106**	0.0200	0.1041	0.0816	0.1099
	STD	0.0096	0.0111	0.0205	0.0232	0.0300
SpectEW	AVG	**0.1273**	0.1498	0.1434	0.1532	0.1427
	STD	0.0162	0.0100	0.0162	0.0176	0.0208
CongressEW	AVG	**0.0094**	0.0167	0.0283	0.0324	0.0702
	STD	0.0078	0.0040	0.0101	0.0134	0.0149
IonosphereEW	AVG	0.0795	**0.0122**	0.0988	0.1272	0.1237
	STD	0.0131	0.0095	0.0128	0.0108	0.0186
KrvskpEW	AVG	0.0362	**0.0262**	0.0514	0.0994	0.0933
	STD	0.0089	0.0033	0.0089	0.0488	0.0394
Tic-tac-toe	AVG	0.1839	**0.1594**	0.1767	0.2318	0.2319
	STD	0.0194	0.0063	0.0081	0.0243	0.0218
Vote	AVG	0.0432	**0.0126**	0.0476	0.0385	0.0627
	STD	0.0086	0.0073	0.0088	0.0090	0.0175
WaveformEW	AVG	0.2563	**0.2453**	0.2373	0.2514	0.2508
	STD	0.0096	0.0070	0.0080	0.0133	0.0164
WineEW	AVG	0.0060	**0.0027**	0.0446	0.0091	0.0255
	STD	0.0052	0.0005	0.0169	0.0117	0.0171
Zoo	AVG	0.0226	**0.0028**	0.0066	0.0046	0.0521
	STD	0.0224	0.0003	0.0095	0.0008	0.0320
Ranking	W|T|L	4|0|14	14|0|4	0|0|18	0|0|18	0|0|18

30

Table 15: Comparison between the best BDA V-shaped and BDA S-shaped time-dependent variants and other meta-heuristics in term of specificity.

Benchmark	Measure	BDA		bGWO	BGSA	BBA
		TV_{V4}	TV_{S3}			
Breastcancer	AVG	0.9540	**1.0000**	0.9689	0.8935	0.9091
	STD	0.0190	0.0000	0.0184	0.0460	0.1239
BreastEW	AVG	**0.9595**	0.9470	0.9304	0.8959	0.8535
	STD	0.0189	0.0065	0.0191	0.0314	0.0868
Exactly	AVG	0.8821	**1.0000**	0.7478	0.5164	0.3320
	STD	0.1081	0.0000	0.1313	0.2363	0.1571
Exactly2	AVG	**0.4284**	0.3296	0.2950	0.3176	0.2370
	STD	0.0868	0.1692	0.1076	0.0907	0.1135
HeartEW	AVG	**0.8606**	0.8122	0.7460	0.8238	0.7333
	STD	0.0506	0.0185	0.0539	0.1017	0.1028
M-of-n	AVG	0.9705	**1.0000**	0.9213	0.8818	0.7940
	STD	0.0299	0.0000	0.0482	0.0524	0.0918
SonarEW	AVG	**1.0000**	0.9949	0.9517	0.8580	0.8236
	STD	0.0000	0.0133	0.0404	0.0653	0.0779
SpectEW	AVG	0.9356	**0.9667**	0.8894	0.8148	0.9397
	STD	0.0229	0.0307	0.0453	0.0651	0.0458
CongressEW	AVG	0.9941	**0.9977**	0.9441	0.9422	0.8204
	STD	0.0120	0.0087	0.1123	0.0503	0.1182
IonosphereEW	AVG	0.8396	**0.9769**	0.7667	0.7529	0.7027
	STD	0.0281	0.0259	0.0571	0.0521	0.0601
KrvskpEW	AVG	0.9649	**0.9732**	0.9240	0.8904	0.7775
	STD	0.0115	0.0053	0.0200	0.0522	0.1083
Tic-tac-toe	AVG	0.6781	**0.7738**	0.6628	0.6348	0.5068
	STD	0.0325	0.0329	0.0469	0.0767	0.1147
Vote	AVG	0.9496	**0.9838**	0.9260	0.9504	0.8514
	STD	0.0163	0.0126	0.0283	0.0164	0.0695
Ranking	W|T|L	4|0|9	9|0|4	0|0|13	0|0|13	0|0|13

Table 16: Comparison between the best BDA V-shaped and BDA S-shaped time-dependent variants and other meta-heuristics in term of Sensitivity.

Benchmark	Measure	BDA		bGWO	BGSA	BBA
		TV_{V4}	TV_{S3}			
Breastcancer	AVG	**0.9900**	0.9886	0.9819	0.9707	0.9471
	STD	0.0094	0.0000	0.0134	0.0166	0.0185
BreastEW	AVG	0.9824	**0.9960**	0.9184	0.9528	0.9425
	STD	0.0146	0.0071	0.0259	0.0214	0.0288
Exactly	AVG	0.9472	**1.0000**	0.8782	0.8316	0.7632
	STD	0.0389	0.0000	0.0606	0.0820	0.0652
Exactly2	AVG	0.8356	**0.9322**	0.7904	0.7676	0.8125
	STD	0.0240	0.0344	0.0606	0.0732	0.0698
HeartEW	AVG	0.9031	**0.9556**	0.8373	0.7354	0.7253
	STD	0.0388	0.0463	0.0481	0.1034	0.0827
M-of-n	AVG	0.9784	**1.0000**	0.8991	0.7390	0.5900
	STD	0.0283	0.0000	0.0668	0.0856	0.1536
SonarEW	AVG	**0.9905**	0.9667	0.8273	0.8778	0.8019
	STD	0.0194	0.0317	0.0613	0.0744	0.0739
SpectEW	AVG	0.5815	0.2778	0.5033	**0.6333**	0.1111
	STD	0.0997	0.1974	0.1066	0.2920	0.1185
CongressEW	AVG	**0.9956**	0.9810	0.9541	0.9428	0.9026
	STD	0.0081	0.0053	0.0292	0.0315	0.0800
IonosphereEW	AVG	0.9949	**0.9993**	0.9630	0.9484	0.9551
	STD	0.0124	0.0041	0.0229	0.0273	0.0279
KrvskpEW	AVG	0.9762	**0.9853**	0.9448	0.9058	0.8014
	STD	0.0097	0.0048	0.0199	0.0626	0.0868
Tic-tac-toe	AVG	**0.8984**	0.8843	0.8723	0.8304	0.7425
	STD	0.0298	0.0072	0.0290	0.0460	0.1050
Vote	AVG	0.9922	**1.0000**	0.9684	0.9302	0.8623
	STD	0.0255	0.0000	0.0528	0.0748	0.1787
Ranking	W|T|L	4|0|9	8|0|5	0|0|13	1|0|12	0|0|13

BDA-based approaches beat other optimizers for all cases. For 69.23% of test cases, TV_{V4} provides the highest AUC rates. It is also observed that TV_{S3} outperforms other methods for the other problems.

Table 17: Comparison between the best BDA V-shaped and BDA S-shaped time dependent variants and other meta-heuristics in term of AUC.

Benchmark	Measure	BDA		bGWO	BGSA	BBA
		TV_{V4}	TV_{S3}			
Breastcancer	AVG	0.9720	**0.9943**	0.9754	0.9321	0.9281
	STD	0.0069	0.0000	0.0108	0.0266	0.0637
BreastEW	AVG	0.9710	**0.9715**	0.9244	0.9244	0.8980
	STD	0.0100	0.0044	0.0136	0.0150	0.0459
Exactly	AVG	0.9147	**1.0000**	0.8130	0.6740	0.5476
	STD	0.0704	0.0000	0.0890	0.1533	0.0710
Exactly2	AVG	**0.6320**	0.6309	0.5427	0.5426	0.5248
	STD	0.0381	0.0685	0.0471	0.0340	0.0400
HeartEW	AVG	0.8819	**0.8839**	0.7917	0.7796	0.7293
	STD	0.0337	0.0168	0.0387	0.0637	0.0620
M-of-n	AVG	0.9744	**1.0000**	0.9102	0.8104	0.6920
	STD	0.0270	0.0000	0.0539	0.0646	0.0908
SonarEW	AVG	**0.9952**	0.9808	0.8895	0.8679	0.8127
	STD	0.0097	0.0144	0.0397	0.0460	0.0580
SpectEW	AVG	**0.7585**	0.6222	0.6964	0.7241	0.5254
	STD	0.0441	0.0840	0.0427	0.1222	0.0586
CongressEW	AVG	**0.9949**	0.9894	0.9491	0.9425	0.8615
	STD	0.0076	0.0049	0.0576	0.0282	0.0837
IonosphereEW	AVG	0.9172	**0.9881**	0.8648	0.8506	0.8289
	STD	0.0141	0.0128	0.0304	0.0285	0.0325
KrvskpEW	AVG	0.9705	**0.9792**	0.9344	0.8981	0.7894
	STD	0.0088	0.0031	0.0190	0.0529	0.0900
Tic-tac-toe	AVG	0.7883	**0.8290**	0.7676	0.7326	0.6247
	STD	0.0188	0.0128	0.0317	0.0425	0.0796
Vote	AVG	0.9709	**0.9919**	0.9472	0.9403	0.8568
	STD	0.0098	0.0063	0.0253	0.0354	0.1150
Ranking	W|T|L	4|0|9	9|0|4	0|0|13	0|0|13	0|0|13

The results thus far compared the average performance, STD, best and worst performance of algorithms for 30 runs. To judge whether results are significant or not, the Wilcoxon statistical test with 5% significance was conducted [68]. The p-values of the Wilcoxon test based on fitness values are reported in Table 18. Such statistical tests consider all runs and can verify that the observed differences and improvements are significantly meaningful. Table 18 shows that the superiority of TV_{S3} over bGWO, BGSA, and BBA is statistically significant for all cases. In addition, the observed differences between TV_{S3} and TV_{V4} are statistically significant for most cases.

The p-values of the Wilcoxon test according to the number of features are shown in Table 19. As per results in Table 19, the observed differences between the TV_{V4} and TV_{S3} techniques are statistically meaningful for all problems. Based on Tables 13 and 19, it can be detected that BBA optimizer has obtained better results for several problems but the observed differences are not statistically significant. Hence, it is not significantly better

Table 18: p-values of the Wilcoxon test for the classification accuracy results of TV_{S3} and other algorithms (p $\geq$ 0.05 are underlined).

Datasets	TV_{V4}	bGWO	BGSA	BBA
Breast Cancer	4.38E-13	3.09E-10	1.03E-12	1.12E-12
BreastEW	<u>1.11E-2</u>	1.03E-11	1.05E-11	1.19E-11
Exactly	5.72E-11	1.15E-12	1.64E-11	1.20E-12
Exactly2	1.46E-9	3.19E-10	2.50E-11	8.65E-10
HeartEW	<u>1.16E-1</u>	2.30E-11	2.99E-9	2.37E-11
Lymphography	6.52E-12	6.65E-12	5.65E-11	7.33E-12
M-of-n	5.75E-9	4.48E-12	1.20E-12	1.20E-12
penglungEW	5.59E-13	2.71E-14	5.25E-6	7.73E-13
SonarEW	3.08E-4	1.04E-11	2.53E-11	1.07E-11
SpectEW	4.57E-8	2.68E-7	1.42E-10	9.02E-12
CongressEW	2.53E-6	2.86E-8	3.98E-11	4.95E-12
IonosphereEW	1.27E-11	1.54E-11	1.54E-11	1.57E-11
KrvskpEW	1.59E-6	2.70E-11	4.60E-10	2.72E-11
Tic-tac-toe	2.35E-10	1.29E-12	3.62E-12	3.07E-12
Vote	3.38E-12	2.47E-11	2.38E-11	1.18E-11
WaveformEW	1.43E-4	2.95E-11	2.97E-11	2.97E-11
WineEW	<u>3.34E-1</u>	1.13E-11	1.17E-5	1.14E-12
Zoo	2.85E-4	8.15E-2	8.15E-2	1.40E-11

than BDA-based versions. On the other hand, referring to Tables 12 and 18, we see that both variants of BDA can outperform BBA in term of accuracy for all problems and all differences are statistically significant.

Boxplots of accuracy results for BDA V-shaped and BDA S-shaped time dependent variants versus other competitors on the compared datasets are shown in Figs. 7 and 7.

6.7. Comparison with other meta-heuristics

This section compares the proposed TV-BDA approaches with popular FS methods proposed in previous studies in terms of classification accuracy rates. The results of TV_{S3} and TV_{V4} are compared with the results of GA and PSO from [42] and the results of the bGWO1, bGWO2, GA, and PSO from [53] in Table 20. It is worth mentioning that the results of the first GA and PSO versions where executed using the source code from the authors in [42], while the results of bGWO1, bGWO2, GA, and PSO where obtained from the paper [53], where the same datasets are used.

In table 20, the substantial superiority of the proposed approaches is observed, where TV_{S3} is capable of revealing the best results for fourteen datasets, and TV_{V4} has discovered the best results for four datasets. These results show again how TV-TFs influences the effectiveness and results of the BDA by enhancing its exploration and exploitation capabilities, to find the most informative features that provides the maximum classification accuracy for different datasets in different dimensions. The key reason for this excellent performance is that the proposed time-varying transfer mechanisms can provide more variety for exploration and exploitation, which leads to improved classification rates. The results are also visually compared in Fig. 9. It can be seen that the classification rates of TV_{S3} and TV_{V4} are relatively higher (shown using darker colors) than those attained by
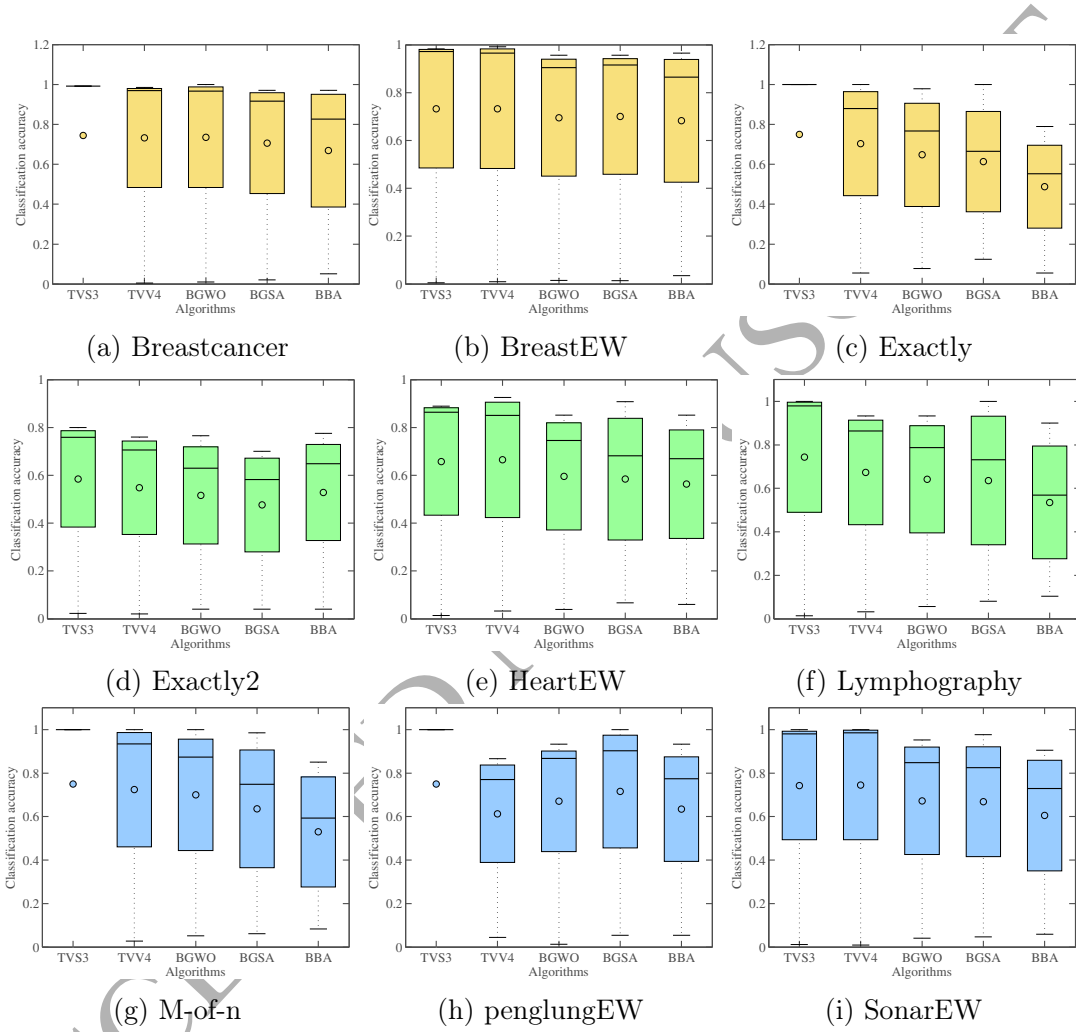
34

Figure 7: Boxplots of accuracy results for BDA V-shaped and BDA S-shaped time dependent variants versus other competitors on Breastcancer, BreastEW, Exactly, Exactly2, HeartEW, Lymphography, M-of-n, penglungEW, and SonarEW datasets
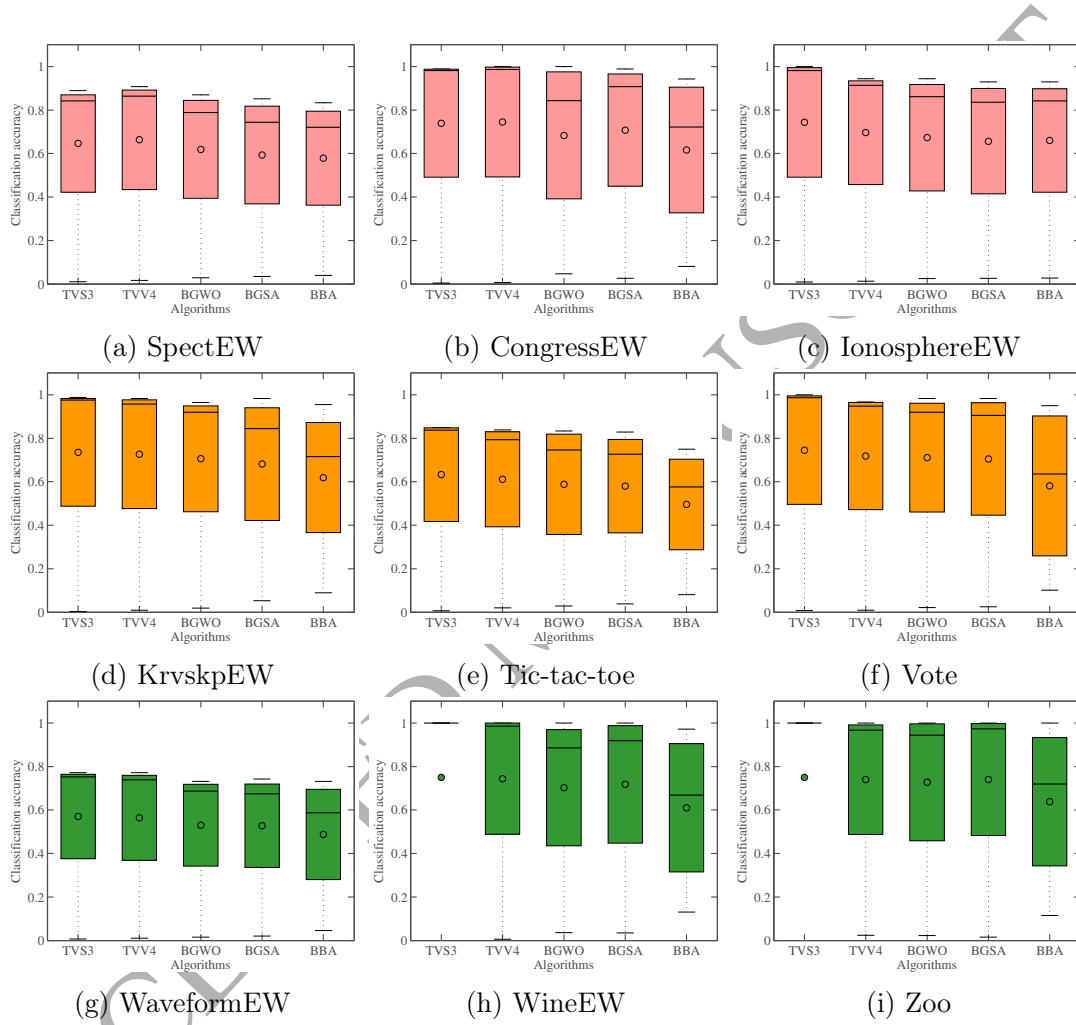
Figure 8: Boxplots of accuracy results for BDA V-shaped and BDA S-shaped time dependent variants versus other competitors on SpectEW, CongressEW, IonosphereEW, KrvskpEW, Tic-tac-toe, Vote, WaveformEW, WineEW, and Zoo datasets

36

Table 19: p-values of the Wilcoxon test for the number of features obtained by TV_{S3} versus other algorithms (p $\geq$ 0.05 are underlined).

Datasets	TV_{V4}	bGWO	BGSA	BBA
Breast Cancer	1.19E-02	9.76E-06	6.24E-03	2.26E-05
BreastEW	5.48E-09	6.51E-09	3.23E-07	5.61E-01
Exactly	8.53E-13	2.46E-11	1.09E-07	7.61E-03
Exactly2	2.75E-04	8.15E-01	2.74E-01	2.01E-01
HeartEW	9.14E-09	4.29E-09	5.81E-02	1.62E-01
Lymphography	4.84E-05	1.41E-03	4.49E-01	2.91E-03
M-of-n	1.75E-12	1.46E-12	7.38E-12	2.86E-01
penglungEW	2.94E-11	2.90E-11	2.88E-11	1.90E-02
SonarEW	1.00E-08	1.16E-07	9.60E-02	1.75E-01
SpectEW	8.73E-09	6.41E-10	2.87E-05	7.07E-03
CongressEW	8.51E-03	8.44E-09	1.89E-02	6.42E-01
IonosphereEW	8.26E-10	1.84E-09	1.41E-05	3.10E-01
KrvskpEW	9.42E-08	3.37E-11	1.35E-01	3.42E-09
Tic-tac-toe	5.80E-05	4.93E-07	3.18E-09	3.71E-10
Vote	1.05E-10	7.56E-10	9.94E-10	2.31E-06
WaveformEW	1.03E-08	4.30E-10	1.26E-01	7.84E-08
WineEW	9.47E-11	2.50E-11	3.07E-11	2.49E-03
Zoo	1.22E-11	2.31E-09	3.10E-10	5.55E-06

other approaches for several datasets. The golden areas indicates rates in the [60% 80%] interval. It can be observed that the proposed approaches (TV_{S3} and TV_{V4}) have smaller golden areas, which visually shows the result improvement.

By reconsidering all the presented results, we can summarize observations. Time-varying TFs can effectively improve the BDA to generally deliver improved results compared to the versions based on traditional TFs. The dynamic nature of TV-based functions can assist the proposed binary optimizer to search the global feature space in a time-aware manner that focus around more promising neighborhood solutions when it is necessary. However, the current non-time-varying nature of TFs cannot adapt the exploration behavior at the beginning of the optimization process when it is required to deal with challenging feature spaces. Thus, it was seen that some promising regions inside the feature space remained unexplored or are not explored enough. This shows the significant role of dynamically-varying TFs to enhance the exploration and exploitation trends of the BDA. The dynamic behavior of proper TFs has mitigated the immature convergence and stagnation behaviors of the basic BDA with non-dynamic TFs. The reason is that in the case of stagnation, the proposed BDA can change its transfer function in the next iterations to avoid LO and it has thus an increased chance of escaping from them by fine-tuning its exploration and exploitation behaviors for the rest of the steps.

The key reason behind all observations is the benefits provided using TV-based TFs in efficiently balancing exploration and exploitation. Changing the shape of TF proportionally to the iteration counter (time dimension) allows the algorithm to leverage the impact of solid TFs on exploration and exploitation by using more varied searching patterns. Different shapes of TFs have varied the probability values that directly translate to the way that the BDA provides diversity or intensity around candidate features.

Table 20: Comparison between the best BDA V-shaped and BDA S-shaped time dependent variants and other meta-heuristics from the literature in terms of accuracy

Benchmark	BDA		GA1 [42]	PSO1 [42]	bGWO1 [53]	bGWO2 [53]	GA2 [53]	PSO2 [53]
	TV_{V4}	TV_{S3}						
Breastcancer	0.9771	**0.9929**	0.957	0.949	0.976	0.975	0.968	0.967
BreastEW	0.9740	**0.9792**	0.923	0.933	0.924	0.935	0.939	0.933
Exactly	0.9290	**1.0000**	0.822	0.973	0.708	0.776	0.674	0.688
Exactly2	0.7257	**0.7725**	0.677	0.666	0.745	0.750	0.746	0.730
HeartEW	**0.8858**	0.8759	0.732	0.745	0.776	0.776	0.780	0.787
Lymphography	0.8950	**0.9922**	0.758	0.759	0.744	0.700	0.696	0.744
M-of-n	0.9732	**1.0000**	0.916	0.996	0.908	0.963	0.861	0.921
penglungEW	0.8067	**1.0000**	0.672	0.879	0.600	0.584	0.584	0.584
SonarEW	**0.9952**	0.9841	0.833	0.804	0.731	0.729	0.754	0.737
SpectEW	**0.8765**	0.8519	0.756	0.738	0.820	0.822	0.793	0.822
CongressEW	**0.9950**	0.9866	0.898	0.937	0.935	0.938	0.932	0.928
IonosphereEW	0.9249	**0.9911**	0.863	0.876	0.807	0.834	0.814	0.819
KrvskpEW	0.9705	**0.9794**	0.940	0.949	0.944	0.956	0.920	0.941
Tic-tac-toe	0.8215	**0.8469**	0.764	0.750	0.728	0.727	0.719	0.735
Vote	0.9617	**0.9894**	0.808	0.888	0.912	0.920	0.904	0.904
WaveformEW	0.7488	0.7580	0.712	0.732	0.786	**0.789**	0.773	0.762
WineEW	0.9991	**1.0000**	0.947	0.937	0.930	0.920	0.937	0.933
Zoo	0.9825	**1.0000**	0.946	0.963	0.879	0.879	0.855	0.861
Ranking (W|T|L)	**4|0|14**	**13|0|5**	0|0|18	0|0|18	0|0|18	1|0|17	0|0|18	0|0|18
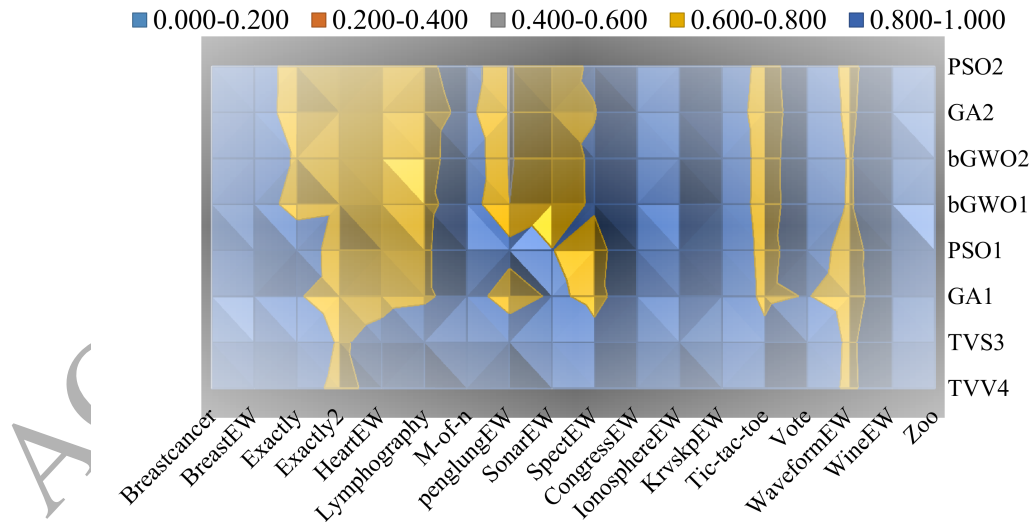
Figure 9: Comparison between TVV4, TVS3 and other techniques from the literature in terms of classification accuracy. Note that better results are indicated in dark blue.

Another main observation was that S-shaped functions outperform V-shaped functions. This is due to the mechanism that each family uses to update the position. When using an S-shaped TFs, the position will be updated using Eq. 11, in which the search agents are required to change the values of 1 to 0 if the step vector level is high. In contrast, the updating rules in the V-shaped TFs are different. The search agents will not be forced to take the values of 0 or 1, but they encourage search agents to stay in their current positions when their step vector values are low or switch to their complements when the step vector values are high. This can significantly degrade exploration of a metaheuristic algorithm.

According to the no free lunch (NFL) theorems [69], we cannot propose a universal best optimizer or feature selection approach. Hence, the proposed wrapper techniques follow the NFL rule.

6.8. Results of algorithms on high-dimensional small instances datasets

In some fields such as the medical and biological studies, it is a hard task to get new instances frequently, since some experiments may take a long time to be reproduced. However, it is well-known that in such fields the number of features to be assessed are very huge, e,g., a dataset may contain thousands or even millions of features. In this case, a dataset may contain a large number of features while the number of instances is relatively very small.

For FS methods, it is a big challenge to deal with datasets that contain a large number of features, or a few number of instances due to two main reasons: the low number of instances (examples) is insufficient to train the learning model, and the large number of features increases the search space where the heuristic approach cannot explore most target regions [70].

In the previous sections, the reported results revealed the capabilities of the proposed approach in dealing with several datasets (with features from 9 to 325) with much success. In this subsection, new experiments are conducted using three well-known medical datasets [71].

The datasets are listed in Table 21. As can be seen in Table 21, three multi-class datasets with a huge number of features and a few number of samples are adopted.

Table 21: Details of high-dimensional small instances datasets [71]

Dataset	No. of Attributes	No. of Objects	No. of Classes
Brain_Tumor2	10367	50	4
SRBCT	2308	83	4
9_Tumors	5726	60	9

In this section, we are interested to compare the best approaches among the proposed ones (i.e., TV_{S3} and TV_{V4}) against other methods (i.e., bGWO, BGSA and BBA) in terms of classification accuracy and fitness values, which both AVG and STD values are reported in Tables 22 and 23. Note that the tabulated results for each algorithm are the average of 30 independent runs.

39

Inspecting the results in Table 22, we can observe that TV_{V4} has obtained the best classification accuracy result among all approaches in one dataset (namely: SRBCT) out of three, while the TV_{S3} approach achieved the best results on two datasets (Brain_Tumor2 and 9_Tumor). It is worth mentioning that the difference between the accuracy obtained by TV_{S3} and the other approaches varies from 15% to 26% on the Brain_Tumor2. Moreover, on the 9_Tumor, TV_{S3} is better than bGWO by 17%, and bGSA by 4%, but BBA has attained a very low classification accuracy. TV_{V4} is still better than other approaches with lower difference in classification accuracy.

From Table 22, the STD values of the proposed approaches are low which indicates how much the values are close to the mean value. This indicates the satisfactory stability of the proposed approach and its capabilities in searching the promising regions of the search space. Boxplots of accuracy results are also shown in Fig. 10.

Table 22: Comparison between the best BDA V-shaped and BDA S-shaped time dependent variants and other meta-heuristics in term of accuracy

Benchmark	Measure	BDA		bGWO	BGSA	BBA
		TVV4	TVS3			
Brain_Tumor2	AVG	0.6000	**0.7100**	0.5600	0.5133	0.4500
	STD	0.0000	0.0305	0.0498	0.0346	0.0572
SRBCT	AVG	**0.9020**	0.8275	0.8882	0.9000	0.8667
	STD	0.0520	0.0435	0.0357	0.0441	0.0265
9_Tumors	AVG	0.5028	**0.5611**	0.3917	0.5228	0.1033
	STD	0.0152	0.0375	0.0543	0.0285	0.0394

(a) Brain_Tumor2   (b) SRBCT   (c) 9_Tumors
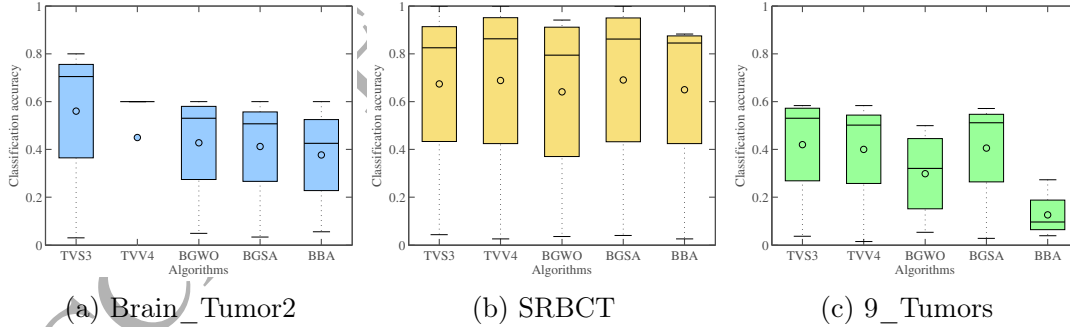
Figure 10: Boxplots of accuracy results for BDA V-shaped and BDA S-shaped time dependent variants versus other optimizers on Brain_Tumor2, SRBCT, and 9_Tumors datasets

Table 23 compares the fitness values and the associated STD values for TV_{S3} and TV_{V4} versus other algorithms. From Table 23, it is clear that both approaches (TV_{S3} and TV_{V4}) obtained the best fitness values with very competitive STD values over all datasets. The number of selected features for all approaches are presented in Table 24. The results shows that bGWO obtained the smallest number of features but it obtained much lower classification accuracy than the proposed approaches. This can be seen in the fitness values

40

(that merge both classification accuracy and the number of selected features), where bGWO obtained the worst fitness values compared to proposed approaches. Although BBA obtained the lowest number of features in one dataset, it has a very low classification rate.

Table 23: Comparison between the best BDA V-shaped and BDA S-shaped time dependent variants and other meta-heuristics in term of fitness results

Benchmark	Measure	BDA		bGWO	BGSA	BBA
		TVV4	TVS3			
Brain_Tumor2	AVG	0.4010	**0.2920**	0.4560	0.3018	0.3414
	STD	0.0000	0.0302	0.0495	0.0000	0.0493
SRBCT	AVG	**0.0224**	0.0535	0.1297	0.0608	0.0609
	STD	0.0271	0.0220	0.0309	0.0106	0.0106
9_Tumors	AVG	0.4970	**0.4393**	0.5633	0.4564	0.7948
	STD	0.0151	0.0371	0.0416	0.0337	0.0278

Table 24: Comparison between the best BDA V-shaped and BDA S-shaped time dependent variants and other meta-heuristics in terms of minimum number of features

Benchmark	Measure	BDA		bGWO	BGSA	BBA
		TVV4	TVS3			
Brain_Tumor2	AVG	5134.23	5121.43	**4150.90**	4932.43	4992.90
	STD	40.78	33.11	509.29	24.48	60.39
SRBCT	AVG	1135.63	1135.57	**907.90**	1047.23	1068.33
	STD	29.31	18.78	78.88	12.20	15.37
9_Tumors	AVG	2715.17	2758.03	2914.27	2859.13	**2318.67**
	STD	17.20	36.38	300.60	40.62	186.95

The p-values of the Wilcoxon test are tabulated in Table 25. These values can verify that the observed differences and improvements are significantly meaningful for all cases.

Table 25: p-values of the Wilcoxon test for the fitness results of TV_{V4} versus other algorithms

	TV_{S3}	bGWO	BGSA	BBA
Brain_Tumor2	1.57E-10	9.96E-08	2.79E-11	2.84E-11
SRBCT	8.48E-06	6.70E-06	1.91E-03	3.46E-09
9_Tumors	4.39E-05	2.91E-11	4.44E-04	2.99E-11

7. Conclusion and future directions

In this paper, the performance of the DA was improved using different TFs to convert the step vector from continuous to a binary space. Eight different transfer functions that belong to two groups (S-shaped and V-shaped) were employed to investigate their

effectiveness on the basic BDA. The main contribution was the proposal of time-varying S-shaped and V-shaped transfer functions to leverage the impact of the step vector on balancing the exploration and exploitation behavior. A set of well-known FS datasets from the UCI data repository were used to evaluate the proposed approach, and the results were compared with the results from other state-of-the-art algorithms. The experimental results showed the superior performance for the time-varying S-shaped BDA approaches compared with other investigated approaches. The discussions and the extensive analyses of the results revealed that time-varying transfer functions can be utilized as an effective way of improving the exploration and exploitation behavior of the BDA for feature selection tasks.

This research opens several research directions for future work in the fields of optimization, metaheuristics, feature selection and applications of these disciplines. As future directions, we think that proposing several new time-dependent TFs is highly beneficial to develop enhanced binary optimizers, and may change the direction of research in the binary optimization field. As a next step, developing new versions of DA by proposing new operators in binary space can also be an interesting research direction. Finally, the proposed binary BDA approaches can be applied as preprocessing step of many pattern recognition, machine learning and feature selection tasks.

8. Acknowledgements

References

[1] M. Eshtay, H. Faris, N. Obeid, Improving extreme learning machine by competitive swarm optimization and its application for medical diagnosis problems, Expert Systems with Applications 104 (2018) 134 – 152.

[2] I. Zelinka, A survey on evolutionary algorithms dynamics and its complexity–mutual relations, past, present and future, Swarm and Evolutionary Computation 25 (2015) 2–14.

[3] A. A. Heidari, R. A. Abbaspour, A. R. Jordehi, Gaussian bare-bones water cycle algorithm for optimal reactive power dispatch in electrical power systems, Applied Soft Computing 57 (2017) 657–671.

[4] A. A. Heidari, R. A. Abbaspour, A. R. Jordehi, An efficient chaotic water cycle algorithm for optimization tasks, Neural Computing and Applications 28 (2017) 57–85.

[5] A. A. Heidari, H. Faris, I. Aljarah, S. Mirjalili, An efficient hybrid multilayer perceptron neural network with grasshopper optimization, Soft Computing (2018) 1–18.

[6] S. Mirjalili, Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems, Neural Computing and Applications 27 (2016) 1053–1073.

[7] R. H. Bhesdadiya, M. H. Pandya, I. N. Trivedi, N. Jangir, P. Jangir, A. Kumar, Price penalty factors based approach for combined economic emission dispatch problem solution using dragonfly algorithm, in: 2016 International Conference on Energy Efficient Technologies for Sustainability (ICEETS), pp. 436–441.

[8] V. Suresh, S. Sreejith, Generation dispatch of combined solar thermal systems using dragonfly algorithm, Computing 99 (2017) 59–80.

[9] M. Hamdy, A.-T. Nguyen, J. L. M. Hensen, A performance comparison of multi-objective optimization algorithms for solving nearly-zero-energy-building design problems, Energy and Buildings 121 (2016) 57–71.

[10] P. T. Daely, S. Y. Shin, Range based wireless node localization using dragonfly algorithm, in: Ubiquitous and Future Networks (ICUFN), 2016 Eighth International Conference on, IEEE, pp. 1012–1015.

[11] E. Elhariri, N. El-Bendary, A. E. Hassanien, Bio-inspired optimization for feature set dimensionality reduction, in: Advances in Computational Tools for Engineering Applications (ACTEA), 2016 3rd International Conference on, IEEE, pp. 184–189.

[12] M. A. Salam, H. M. Zawbaa, E. Emary, K. K. A. Ghany, B. Parv, A hybrid dragonfly algorithm with extreme learning machine for prediction, in: 2016 International Symposium on INnovations in Intelligent SysTems and Applications (INISTA), pp. 1–6.

[13] G. Raman, G. Raman, C. Manickam, S. I. Ganesan, Dragonfly algorithm based global maximum power point tracker for photovoltaic systems, in: International Conference in Swarm Intelligence, Springer, pp. 211–219.

[14] C. Hema, S. Sankar, et al., Energy efficient cluster based protocol to extend the rfid network lifetime using dragonfly algorithm, in: Communication and Signal Processing (ICCSP), 2016 International Conference on, IEEE, pp. 0530–0534.

[15] M. Abdel-Basset, Q. Luo, F. Miao, Y. Zhou, Solving 0–1 knapsack problems by binary dragonfly algorithm, in: International Conference on Intelligent Computing, Springer, pp. 491–502.

[16] R. Bhesdadiya, M. H. Pandya, I. N. Trivedi, N. Jangir, P. Jangir, A. Kumar, Price penalty factors based approach for combined economic emission dispatch problem solution using dragonfly algorithm, in: Energy Efficient Technologies for Sustainability (ICEETS), 2016 International Conference on, IEEE, pp. 436–441.

[17] S. R. KS, S. Murugan, Memory based hybrid dragonfly algorithm for numerical optimization problems, Expert Systems with Applications 83 (2017) 63–78.

[18] J. Song, S. Li, Elite opposition learning and exponential function steps-based dragonfly algorithm for global optimization, in: Information and Automation (ICIA), 2017 IEEE International Conference on, IEEE, pp. 1178–1183.

[19] M. Mafarja, D. Eleyan, I. Jaber, A. Hammouri, S. Mirjalili, Binary dragonfly algorithm for feature selection, in: New Trends in Computing Sciences (ICTCS), 2017 International Conference on, IEEE, pp. 12–17.

[20] S. Mirjalili, A. Lewis, S-shaped versus v-shaped transfer functions for binary particle swarm optimization, Swarm and Evolutionary Computation 9 (2013) 1–14.

[21] J. C. Bansal, K. Deep, A modified binary particle swarm optimization for knapsack problems, Applied Mathematics and Computation 218 (2012) 11042–11061.

[22] M. J. Islam, X. Li, Y. Mei, A time-varying transfer function for balancing the exploration and exploitation ability of a binary pso, Applied Soft Computing 59 (2017) 182–196.

[23] L. Wang, X. Wang, J. Fu, L. Zhen, A novel probability binary particle swarm optimization algorithm and its application, Journal of software 3 (2008) 28–35.

[24] L. Ma, M. Li, Y. Gao, T. Chen, X. Ma, L. Qu, A novel wrapper approach for feature selection in object-based image classification using polygon-based cross-validation, IEEE Geoscience and Remote Sensing Letters 14 (2017) 409–413.

[25] V. Rodriguez-Galiano, J. Luque-Espinar, M. Chica-Olmo, M. Mendes, Feature selection approaches for predictive modelling of groundwater nitrate pollution: An evaluation of filters, embedded and wrapper methods, Science of The Total Environment 624 (2018) 661–672.

[26] Y. Zhu, J. Liang, J. Chen, Z. Ming, An improved nsga-iii algorithm for feature selection used in intrusion detection, Knowledge-Based Systems 116 (2017) 74–85.

[27] K. Ravi, V. Ravi, A novel automatic satire and irony detection using ensembled feature selection and data mining, Knowledge-Based Systems 120 (2017) 15–33.

[28] W.-P. Lee, C.-T. Chen, J.-Y. Huang, J.-Y. Liang, A smartphone-based activity-aware system for music streaming recommendation, Knowledge-Based Systems 131 (2017) 70–82.

[29] Y. Zhang, S. Wang, P. Phillips, G. Ji, Binary pso with mutation operator for feature selection using decision tree applied to spam detection, Knowledge-Based Systems 64 (2014) 22–31.

[30] D. Liang, C.-F. Tsai, H.-T. Wu, The effect of feature selection on financial distress prediction, Knowledge-Based Systems 73 (2015) 289–297.

[31] S. F. Crone, S. Lessmann, R. Stahlbock, The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing, European Journal of Operational Research 173 (2006) 781–800.

[32] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, Recent advances and emerging challenges of feature selection in the context of big data, Knowledge-Based Systems 86 (2015) 33–45.

[33] H. Liu, H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, Kluwer Academic Publishers, Boston, 1998.

[34] Z. Zhu, Y. Ong, M. Dash, WrapperâĂŞfilter feature selection algorithm using a memetic framework, IEEE Transactions on Systems, Man, and Cybernetics 37 (2007) 70–76.

[35] R. Jensen, Q. Shen, Computational Intelligence and Feature Selection: Rough and Fuzzy Approaches, Wiley-IEEE Press, 2008.

[36] E. Talbi, Metaheuristics From design to implementation, Wiley Online Library, 2009.

[37] M. M. Kabir, M. Shahjahan, K. Murase, A new local search based hybrid genetic algorithm for feature selection, Neurocomputing 74 (2011) 2914–2928.

[38] J. Kennedy, R. C. Eberhart, A discrete binary version of the particle swarm algorithm, in: 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, volume 5, pp. 4104–4108.

[39] H. Faris, I. Aljarah, B. Al-Shboul, A Hybrid Approach Based on Particle Swarm Optimization and Random Forests for E-Mail Spam Filtering, Springer International Publishing, Cham, pp. 498–508.

[40] M. Mafarja, R. Jarrar, S. Ahmad, A. Abusnaina, Feature selection using binary particle swarm optimization with time varying inertia weight strategies, in: The 2nd International Conference on Future Networks & Distributed Systems , Amman, Jordan, volume 2, ACM.

[41] M. Mafarja, N. R. Sabar, Rank based binary particle swarm optimisation for feature selection in classification (2018).

[42] S. Kashef, H. Nezamabadi-pour, An advanced aco algorithm for feature subset selection, Neurocomputing 147 (2015) 271–279.

[43] E. Zorarpacı, S. A. Özel, A hybrid approach of differential evolution and artificial bee colony for feature selection, Expert Systems with Applications 62 (2016) 91–103.

[44] Y.-P. Chen, Y. Li, G. Wang, Y.-F. Zheng, Q. Xu, J.-H. Fan, X.-T. Cui, A novel bacterial foraging optimization algorithm for feature selection, Expert Systems with Applications 83 (2017) 1–17.

[45] D. Karaboga, An idea based on honey bee swarm for numerical optimization, Technical Report, Technical report-tr06, Erciyes university, engineering faculty, computer engineering department, 2005.

[46] S. Mirjalili, The ant lion optimizer, Advances in Engineering Software 83 (2015) 80–98.

[47] M. Mafarja, S. Mirjalili, Hybrid binary ant lion optimizer with rough set and approximate entropy reducts for feature selection, Soft Computing (2018).

[48] M. Mafarja, D. Eleyan, S. Abdullah, S. Mirjalili, S-shaped vs. v-shaped transfer functions for ant lion optimization algorithm in feature selection problem, in: Proceedings of the International Conference on Future Networks and Distributed Systems, ACM, p. 14.

[49] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, Advances in Engineering Software 69 (2014) 46–61.

[50] A. A. Heidari, P. Pahlavani, An efficient modified grey wolf optimizer with lévy flight for optimization tasks, Applied Soft Computing 60 (2017) 115–134.

[51] A. A. Heidari, R. A. Abbaspour, Enhanced chaotic grey wolf optimizer for real-world optimization problems: A comparative study, in: Handbook of Research on Emergent Applications of Optimization Algorithms, IGI Global, 2018, pp. 693–727.

[52] E. Emary, H. M. Zawbaa, C. Grosan, A. E. Hassenian, Feature subset selection approach by gray-wolf optimization, in: Afro-European Conference for Industrial Advancement, Springer, pp. 1–13.

[53] E. Emary, H. M. Zawbaa, A. E. Hassanien, Binary grey wolf optimization approaches for feature selection, Neurocomputing 172 (2016) 371–381.

[54] S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, Knowledge-Based Systems 89 (2015) 228–249.

[55] H. M. Zawbaa, E. Emary, B. Parv, M. Sharawi, Feature selection approach based on moth-flame optimization algorithm, in: Evolutionary Computation (CEC), 2016 IEEE Congress on, IEEE, pp. 4612–4617.

[56] H. Faris, M. A. Hassonah, A. M. Al-Zoubi, S. Mirjalili, I. Aljarah, A multi-verse optimizer approach for feature selection and optimizing svm parameters based on a robust system architecture, Neural Computing and Applications (2017).

[57] M. Mafarja, S. Abdullah, Investigating memetic algorithm in solving rough set attribute reduction, International Journal of Computer Applications in Technology 48 (2013) 195–202.

[58] M. Mafarja, S. Mirjalili, Hybrid whale optimization algorithm with simulated annealing for feature selection, Neurocomputing 260 (2017) 302–312.

[59] M. Mafarja, S. Mirjalili, Whale optimization approaches for wrapper feature selection, Applied Soft Computing 62 (2017) 441–453.

[60] H. Faris, M. A. Hassonah, A. M. Al-Zoubi, S. Mirjalili, I. Aljarah, A multi-verse optimizer approach for feature selection and optimizing svm parameters based on a robust system architecture, Neural Computing and Applications (2017).

[61] M. Mafarja, I. Aljarah, A. A. Heidari, A. I. Hammouri, H. Faris, A. Al-Zoubi, S. Mirjalili, Evolutionary population dynamics and grasshopper optimization approaches for feature selection problems, Knowledge-Based Systems 145 (2018) 25 – 45.

[62] I. Aljarah, M. Mafarja, A. A. Heidari, H. Faris, Y. Zhang, S. Mirjalili, Asynchronous accelerating multi-leader salp chains for feature selection, Applied Soft Computing (2018).

[63] C. W. Reynolds, Flocks, herds and schools: A distributed behavioral model, ACM SIGGRAPH computer graphics 21 (1987) 25–34.

[64] S. Mirjalili, G.-G. Wang, L. d. S. Coelho, Binary optimization using hybrid particle swarm optimization and gravitational search algorithm, Neural Computing and Applications (2014) 1–13.

[65] N. S. Altman, An introduction to kernel and nearest-neighbor nonparametric regression, The American Statistician 46 (1992) 175–185.

[66] M. Lichman, UCI machine learning repository, 2013.

[67] H. Faris, I. Aljarah, N. Al-Madi, S. Mirjalili, Optimizing the learning process of feedforward neural networks using lightning search algorithm, International Journal on Artificial Intelligence Tools 25 (2016) 1650033.

[68] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm and Evolutionary Computation 1 (2011) 3–18.

[69] S. Shalev-Shwartz, S. Ben-David, Understanding machine learning: From theory to algorithms, Cambridge university press, 2014.

[70] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, Feature selection for high-dimensional data, Springer, 2015.

[71] A. Statnikov, C. F. Aliferis, I. Tsamardinos, D. Hardin, S. Levy, A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis, Bioinformatics 21 (2004) 631–643.