Dynamic Binary Swordfish Movement Optimization Algorithm for Feature Selection

Faris H. Rizk[1], Khaled Sh. Gaber[2], Marwa M. Eid[3,4*], Doaa Sami Khafaga[5], Amel Ali Alhussan[5], El-Sayed M. El-kenawy[6,7*]

[1]Department of Communications and Electronics, Delta higher Institute of Engineering and Technology, Mansoura 35111, Egypt.
[2]Computer Science and Intelligent Systems Research Center, Blacksburg 24060, Virginia, USA.
[3*]Faculty of Artificial Intelligence, Delta University for Science and Technology, Mansoura, Egypt.
[4*]Jadara University Research Center, Jadara University, Jordan.
[5]Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia.
[6*]Department of Programming, School of Information and Communications Technology (ICT), Bahrain Polytechnic, PO Box 33349, Isa Town, Bahrain.
[7*]Applied Science Research Center. Applied Science Private University, Amman, Jordan.

*Corresponding author(s). E-mail(s): mmm@ieee.org; skenawy@ieee.org;
Contributing authors: faris.hamdi.rizk@gmail.com; khsherif@jcsis.org; dskhafga@pnu.edu.sa; aaalhussan@pnu.edu.sa;

Abstract

Feature selection (FS) is a critical preprocessing stage in machine learning, particularly when dealing with high-dimensional datasets that often contain redundant or irrelevant features. Inefficient handling of such data increases computational complexity and reduces classifier generalization. Existing binary metaheuristics frequently rely on static control parameters, which can lead to premature convergence and poor adaptability.

To address these limitations, this paper proposes the **Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA)**, a novel nature-inspired binary optimizer based on the foraging dynamics of swordfish. DBSMOA extends the original Swordfish Movement Optimization Algorithm (SMOA) by incorporating dynamic behavioral adaptation mechanisms that enable a self-adjusting balance between exploration and exploitation throughout the optimization process. The algorithm employs a sigmoid-based probabilistic binary mapping and dynamically reassigns agent roles through a performance-driven elitist strategy to maintain diversity and stability during search.
Comprehensive evaluations on 52 benchmark datasets show that DBSMOA achieves the lowest average classification error on 35 datasets and the smallest average subset size on 34 datasets, outperforming twelve state-of-the-art binary optimizers, including Binary Particle Swarm Optimization (bPSO), Binary Genetic Algorithm (bGA), and Binary Grey Wolf Optimizer (bGWO). The results demonstrate that DBSMOA delivers competitive accuracy, compact feature subsets, and strong computational efficiency, highlighting its robustness, scalability, and suitability for high-dimensional feature selection and other binary optimization tasks.

Keywords: Dynamic Binary Optimization, Feature Selection, Metaheuristic Algorithms, Swordfish Movement Optimization, high-Dimensional Data Analysis

1 Introduction

The exponential growth of high-dimensional datasets across diverse domains such as medical informatics [1], computational biology [2], fault diagnostics [3], remote sensing [4], and cyber-physical systems [5] has intensified the demand for efficient computational techniques capable of extracting meaningful insights from complex data. In these contexts, datasets often contain hundreds or thousands of features, many of which may be redundant, irrelevant, or noisy. Such superfluous features not only increase computational complexity [6] but also degrade classifier performance through overfitting and reduced generalization ability [7, 8]. Consequently, feature selection (FS)—the task of identifying the most informative subset of features while discarding the rest—has become an indispensable stage in modern data analysis pipelines [9, 10].

Feature selection is inherently a combinatorial optimization problem [11], requiring an exhaustive search over the power set of all possible feature subsets to identify an optimal configuration. The exponential growth of this search space makes exhaustive and deterministic approaches computationally infeasible [12]. As a result, metaheuristic optimization algorithms—methods inspired by natural processes such as biological evolution [13], swarm intelligence [14], and thermodynamics [15]—have emerged as practical tools for tackling large-scale, multimodal optimization problems. These algorithms offer flexible, problem-independent frameworks that can efficiently approximate near-optimal solutions in high-dimensional search spaces [16, 17].

Within this class, binary metaheuristic algorithms have gained considerable attention because feature selection operates in a discrete search domain [18]. In such

frameworks, each candidate solution is encoded as a binary vector, where each bit represents the inclusion (1) or exclusion (0) of a feature. Many well-established continuous metaheuristics have been adapted to binary spaces, including Binary Particle Swarm Optimization (bPSO), Binary Genetic Algorithm (bGA) [19], and Binary Grey Wolf Optimizer (bGWO) [20]. Although these algorithms have demonstrated competitive performance across benchmark problems, they often face challenges such as premature convergence to local optima [21], instability across independent runs, and loss of population diversity—especially in high-dimensional spaces.

A fundamental limitation of many binary metaheuristics lies in their *static search behavior*. Most approaches employ fixed control parameters and movement equations that remain unchanged throughout the optimization process. This rigidity prevents the algorithm from adapting its exploration and exploitation strategies when faced with stagnation or suboptimal convergence trends, often resulting in premature convergence and reduced search diversity. The challenge is further magnified in binary domains, where transition dynamics between discrete states are inherently constrained and sensitive to diversity loss [22].

To address these shortcomings, researchers have increasingly explored *dynamic metaheuristic optimization* strategies that introduce adaptive feedback mechanisms and self-regulating control. In such algorithms, parameters like mutation rates, role assignments, or movement coefficients evolve dynamically in response to population diversity, convergence rate, or fitness variance [23, 24]. This adaptivity enables a continuous adjustment of the balance between exploration and exploitation, resulting in improved robustness, stability, and convergence reliability across problem domains [25]. Dynamic strategies are particularly advantageous for high-dimensional and noisy search spaces, where maintaining adaptability is critical for avoiding stagnation and preserving population diversity.

In addition to these dynamic strategies, recent studies in evolutionary computation have demonstrated the effectiveness of improved and hybridized algorithms that incorporate adaptive control and elite-driven mechanisms. For instance, the Sine Cosine Algorithm guided by elite pool strategy [26] and the Snow Ablation Optimizer [27] have achieved superior global optimization performance through intelligent guidance and natural process simulation. Similarly, multi-strategy and adaptive variants of the Slime Mould Algorithm [28, 29], as well as Q-learning-enhanced JAYA algorithms [30], have showcased the role of reinforcement learning and adaptive parameterization in balancing exploration and exploitation. Furthermore, hybrid models such as the hybrid Grasshopper Optimization Algorithm [31] and the enhanced Social Learning Swarm Optimizer [32] have proven the advantages of combining multiple learning paradigms for robust convergence and faster adaptation. These advancements underscore the growing emphasis on dynamic, hybrid, and self-adaptive strategies for metaheuristic design, motivating the development of DBSMOA as an evolution of this trend.

Despite these advances, most existing dynamic metaheuristics primarily focus on continuous search spaces and do not adequately address the challenges inherent in binary domains, such as abrupt transitions, reduced diversity, and limited adaptability of movement dynamics. This limitation creates a strong motivation to explore biologically inspired swarm behaviors that naturally exhibit collective

intelligence, adaptability, and cooperation—qualities that can be mapped effectively into binary search frameworks. The Swordfish Movement Optimization Algorithm (SMOA) was chosen as the foundation for this work because its underlying behavioral model—coordinated hunting, leader-follower synchronization, and dynamic role switching—offers a natural analogy for balancing global and local search. Unlike static or purely stochastic swarm algorithms, SMOA inherently supports dynamic phase transitions between exploration and exploitation, which can be enhanced further through adaptive binary mechanisms.

Incorporating dynamic behavior into binary optimization allows context-aware modulation of the algorithm's operations [33]. Agents can adjust movement rules according to fitness landscapes, population diversity, or convergence stagnation metrics [34], thereby introducing feedback-driven adaptability and resilience in high-dimensional or noisy environments [35]. Such adaptivity enhances generalization performance by reducing overfitting to spurious feature interactions—an especially beneficial property in biomedical and other small-sample domains [36, 37].

Scalability and computational efficiency are also crucial for practical deployment [38]. As data dimensionality increases in applications such as IoT-enabled health monitoring [39], autonomous systems [40], and large-scale text or image analytics [41], algorithms must deliver both high accuracy and low computational cost [42]. Traditional exhaustive or greedy methods are typically infeasible under such conditions [43], while adaptive, feedback-driven metaheuristics provide a more practical alternative [44]. Moreover, minimizing CPU and memory usage is essential for implementation in resource-constrained environments [45].

Moreover, a recent research direction has sought to enhance the theoretical understanding of metaheuristic algorithms using complex network theory. By modeling the interactions among agents as evolving networks, researchers have uncovered novel insights into convergence dynamics, diversity retention, and information propagation within optimization systems. Notably, works such as "Unlocking New Potentials in Evolutionary Computation with Complex Network Insights" [46] and "Collective Dynamics of Particle Swarm Optimization: A Network Science Perspective" [47] demonstrate how network representations can bridge the gap between empirical performance and theoretical understanding. These perspectives open new avenues for the analytical modeling of algorithms like DBSMOA and will inform future extensions of this work.

Compared with existing dynamic binary algorithms such as Dynamic Binary Butterfly Optimization, Binary Dynamic Grey Wolf Optimization, and adaptive variants of bPSO or bGA, DBSMOA distinguishes itself by employing a dual-level dynamism: (1) adaptive role reallocation among exploration and exploitation agents, and (2) a feedback-driven binary transformation mechanism governed by probabilistic mapping. This dual adaptivity ensures both population diversity and convergence stability, providing a more balanced and responsive search behavior across various high-dimensional datasets.

Building upon these motivations, this study introduces the *Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA)* for feature selection. The proposed algorithm extends the original Swordfish Movement Optimization Algorithm

(SMOA)—designed for continuous spaces—by reformulating its movement, interaction, and selection mechanisms for binary optimization. DBSMOA introduces adaptive dynamics at multiple levels of its structure. The swarm population is divided into exploration and exploitation groups that dynamically alternate roles based on empirical indicators such as agent performance, fitness variance, and iteration progress [48, 49]. In addition, an elitist selection mechanism preserves high-quality solutions across iterations, improving convergence reliability and reducing random degradation [50, 51].

The primary objective of this work is to bridge the gap between accuracy and convergence reliability in binary feature selection [52]. By embedding dual adaptivity—both behavioral and binary—into the optimization process, DBSMOA achieves high flexibility across diverse datasets, including those characterized by large dimensionality, sparsity, and overlapping class boundaries [53]. Thus, DBSMOA represents a biologically inspired, performance-aware optimization framework capable of balancing exploration, exploitation, and computational efficiency in complex binary search spaces [54].

The main contributions of this study are summarized as follows:

- Proposal of a novel metaheuristic algorithm, termed *Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA)*, extending SMOA with dynamic behavioral adaptation and binary search capabilities.
- Development of an adaptive exploration–exploitation control mechanism through real-time population role reallocation guided by fitness-based feedback.
- Integration of a sigmoid-based probabilistic binary mapping for effective discrete-space transformation and balanced search dynamics.
- Introduction of an elitism mechanism to preserve top-performing solutions across generations, enhancing convergence stability and preventing solution degradation.
- DBSMOA provides a dual dynamic adaptation mechanism that differentiates it from existing binary and dynamic algorithms, while leveraging the natural cooperative dynamics of swordfish hunting behavior as a biologically grounded inspiration.
- Comprehensive evaluation on 52 benchmark datasets demonstrating DBSMOA's consistent top-ranking performance in classification accuracy, feature reduction, and computational efficiency.

This study aims to develop a dynamic binary optimizer that adaptively balances exploration and exploitation for robust feature selection. DBSMOA is evaluated against twelve benchmark algorithms across fifty-two datasets to verify its accuracy, convergence behavior, and efficiency. The remainder of this paper is structured as follows: Section 2 reviews related work on feature selection and binary metaheuristics; Section 3 presents the proposed DBSMOA; Section 5 reports experimental results and comparisons; Section 6 provides a detailed analysis; and Section 7 concludes the paper with future research directions.

2 Literature Review

This section surveys prior research relevant to feature selection (FS), organized around the standard categories—Embedded, Wrapper, and Filter—while emphasizing methodological foundations, computational efficiency, and representative applications.

Feature selection reduces dimensionality by identifying informative feature subsets, thereby enhancing model performance, lowering computational cost, and mitigating overfitting. FS methods are generally classified into Embedded, Wrapper, and Filter approaches and, orthogonally, by supervision: supervised, unsupervised, and semi-supervised.

To aid early understanding, Table 1 presents an overview of notable methods, summarizing their core contributions, limitations, and application domains.

Synthesis and Trends (2020–2025)

Table 2 structures recent binary FS research chronologically and thematically. We group methods into three periods reflecting the field's progression: (i) 2020–2021, emphasizing *binary transformations* (S-/V-shaped transfer, angle modulation) to bridge continuous heuristics into FS; (ii) 2022–2023, highlighting *hybridization and multi-strategy designs* (e.g., DE+mutation, swarm fusion) with improved diversity control; and (iii) 2024–2025, demonstrating momentum toward *dynamic, feedback-driven strategies* (time-varying transfer, adaptive local search) with stronger statistical validation. Each row lists representative studies.
From Table 2, three trends are evident: (1) a shift from static transfer functions toward time-varying or feedback-coupled mappings; (2) increased reliance on hybridization (e.g., swarm+DE, chaotic initializations, crisscross/exploration heuristics) to balance global search and local refinement; and (3) broader, more rigorous evaluations (e.g., Wilcoxon tests; multi-dataset UCI/medical/cyber-physical benchmarks). These trends collectively motivate the dynamic design choices in DBSMOA.

2.0.1 Embedded Methods

Embedded FS integrates selection directly into model training, enabling algorithms to learn relevant features while fitting the predictive model. This tight integration promotes computational efficiency and can yield task-specific subsets, especially in high-dimensional settings. The Binary JAYA algorithm (BJAM) [58] adapts the continuous JAYA optimizer for binary spaces using a sinusoidal transfer function and adaptive mutation, showing improvements in accuracy and feature reduction across 22 real-world datasets. The Binary harris hawks Optimizer (hhO) [60] incorporates exploration mechanisms such as greedy selection and Lévy flights in binary search, performing effectively on high-dimensional biomedical datasets with compact, stable subsets.

hybrid and nature-inspired embedded methods have also shown promise. The DE–Sailfish hybrid (DESFO) [61] reported 100% accuracy on certain benchmarks by coupling FS with classifiers such as k-NN and Random Forest. The Golden Lichtenberg Algorithm (GLA) [62] employs Fibonacci-based scaling to enhance search, producing compact feature sets and strong accuracy on UCI datasets. Additional

Table 1 Condensed Summary of Feature Selection Methods in Recent Studies

Method Type	Study Summary	Methodological highlights
Embedded	[55]: Fractal-based EA avoids local minima in optimization tasks.	Uses fractal diffusion for better exploration.
	[56]: GA implementation guidance for FS.	Foundational embedded FS concepts.
	[57]: DTO inspired by bird motion.	Fast, efficient search strategy.
	[58]: BJAM (Binary JAYA) with adaptive mutation.	Boosts FS accuracy via binary tuning.
	[59]: BSBO for Set Covering Problem.	Binary conversion + repair operator.
	[60]: Binary hhO for high-dim FS.	Dynamic, stable wrapper integration.
	[61]: DESFO combines DE and Sailfish.	Achieves 100% accuracy on benchmarks.
	[62]: GLA uses Fibonacci-guided scaling.	Compact features, high accuracy.
Wrapper	[63]: mRMR + firefly for gene FS.	Two-stage hybrid filtering.
	[64]: DE with angle modulation.	Translates continuous FS to binary.
	[65]: bWOA with sigmoid/tanh functions.	FS across 24 datasets.
	[66]: Dynamic sticky BPSO.	Improved search via binary movement.
	[67]: BSCA for power scheduling.	Uses sine/cosine for FS modeling.
	[68]: hBPSOSCA merges BPSO + SCA.	Enhances clustering-based FS.
	[69]: MBO + k-NN; 93% accuracy avg.	Strong global–local balance.
	[70]: BKOA-MUT; FS for 25 UCI sets.	Combines Kepler law + mutation.
	[71]: EDCSA with k-means for FS.	Improved clustering performance.
	[72]: BAOA with sigmoid mapping.	Outperforms BPSO, BGA, BDF.
	[73]: Binary ChOA for gene FS.	Competes with top wrapper methods.
	[74]: BGOA-TVG with Gaussian transfer.	Strong global search; fast convergence.
	[75]: BinCOA with refracted learning and crisscross strategy.	Outperforms 7 wrappers; Wilcoxon test applied.
	[76]: GJO-GWO hybrid FS algorithm.	Multi-strategy fusion with Lagrange interpolation.
	[77]: DBOA using mutation-based local search.	Beats other methods on 20 UCI datasets.
Filter	[78]: iBSSA with local search.	Compact subsets, high accuracy.
	[79]: BEOSA across 22 datasets.	Minimal features, top results.
	[80]: CGSO with chaotic maps.	FS + classifier model simplification.
	[81]: IBAO with local refinement.	Achieves 100% accuracy.
	[82]: BPSO-TEPD with EPD.	Better convergence on FS.
	[83]: AD-PRS-Guided WOA for faults.	97% accuracy in diagnosis.
	[84]: AMBO with memetic logic.	Outperforms 10+ FS algorithms.

Table 2 Chronological–Thematic map of recent binary FS advances (2020–2025)

Period	Dominant Themes	Representative methods and key ideas	FS Role
2020–2021	Binary transformation mechanisms	Angle-modulated DE for binary encoding [64]; S-/V-shaped transfers in bWOA and BAOA [65, 72]; early dynamic binary PSO variants [66]; binary adaptations of scheduling/energy models (BSCA) demonstrating portability to FS [67].	Wrapper (dominant); Filter (emerging)
2022–2023	hybridization & multi-strategy search; diversity preservation	hybrid BPSO–SCA and clustering-oriented FS [68, 71]; Kepler+DE with mutation for robust search [70]; memetic/logic-gate local search [84]; chaotic maps and opposition-based design [75, 80]; strong wrappers with k-NN/SVM backends [69, 81].	Wrapper & Filter (broad); Embedded (select cases)
2024–2025	Dynamic, feedback-driven strategies; time-varying transfers; stronger statistics	Time-varying Gaussian transfer in BGOA-TVG [74]; mutation-based local search in dynamic BOA [77]; elite- and repair-enhanced binary designs [59]; high-accuracy hybrids (DESFO, GLA) demonstrating compactness and stability [61, 62].	Wrapper (with dynamic control); Embedded (hybrids)

contributions include the Fractal-based Evolutionary Algorithm (FEA) [55], which leverages diffusion-inspired search to improve exploration, and the Dipper Throated Optimization (DTO) algorithm [57], which models bird-inspired motion for metaheuristic search. Although not initially developed for FS, these methods provide adaptable frameworks for binary problems. The Satin Bowerbird Optimizer (BSBO) [59] has been applied to discrete optimization (e.g., set covering) using binary conversion and repair, indicating potential utility for FS. Foundational GA work [56] continues to inform embedded FS via encoding, selection, and crossover strategies.
Embedded variants increasingly adopt *lightweight dynamic controls* (e.g., adaptive mutation in BJAM [58], Fibonacci-guided scaling in GLA [62]), aligning with the broader move toward *feedback-aware* search and compact subsets summarized in Table 2.

2.0.2 Wrapper Method

Wrapper FS evaluates feature subsets via predictive performance, typically achieving higher accuracy at the expense of computation. Numerous wrapper strategies use metaheuristics to traverse the subset space. The Chimp Optimization Algorithm (ChOA) [73] adapts to binary domains with S- and V-shaped transfers and crossover to enhance exploration; on biomedical, text, and image datasets, it reports superior accuracy and dimensionality reduction against GA, PSO, and FA. The Enhanced Binary Kepler Optimization Algorithm (BKOA-MUT) [70] combines Keplerian motion with DE mutation, yielding high accuracy and compact subsets across 25 UCI datasets (k-NN/SVM backends). The Improved Binary Aquila Optimization (IBAO) [81] combines global exploration with local refinement, reporting up to 100% accuracy while reducing features by up to 92%.

Further wrapper advances include the Binary Arithmetic Optimization Algorithm (BAOA) [72], which employs sigmoid transfer for binary encoding and outperforms BPSO, BGA, and BDF on 18 UCI datasets (k-NN evaluation). Monarch Butterfly Optimization (MBO) [69], paired with k-NN, reports an average accuracy of 93% across 18 datasets while reducing features. For clustering, EDCSA [71] and

hBPSOSCA [68] demonstrate enhanced unsupervised performance via FS. Angle modulation effectively adapts DE to binary search [64], simplifying binary encoding. Binary variants of Whale Optimization [65] and Sticky PSO [66] improve control over subset size and classification outcomes. The Binary Sine–Cosine Algorithm (BSCA) [67], though developed for unit commitment, shows portability to FS due to its binary transformation and dynamic search behavior. A two-phase mRMR + multilayer binary firefly approach [63] achieves strong classification accuracy and dimensionality reduction on high-dimensional biomedical data.

Recent wrapper work underscores hybridization and dynamic strategies. BGOA-TVG [74] introduces a time-varying Gaussian transfer for mapping continuous to binary spaces, improving exploration–exploitation balance and accuracy on UCI, DEAP, and EPILEPSY datasets compared with traditional S-/V-shaped transfers and several swarm methods. BinCOA [75] tailors Crayfish Optimization to binary search using refracted opposition-based learning and a crisscross strategy, outperforming seven FS methods across 30 benchmarks in accuracy, compactness, and statistical significance. The GJO–GWO hybrid [76] fuses Golden Jackal Optimization with Grey Wolf Optimizer using Lagrange interpolation, consistently achieving lower fitness and higher accuracy. The Dynamic Butterfly Optimization Algorithm (DBOA) [77] augments BOA with a mutation-based local search (LSAM) to address stagnation and diversity issues, showing superior performance on 20 UCI datasets.
Wrapper methods increasingly move from *static* S-/V-shaped transfers [65, 72] to *time-varying/adaptive* mechanisms [74, 77], often within hybrid pipelines [75, 76], as summarized in Table 2.

2.0.3 Filter Method

Filter FS ranks features independently of a predictive model and is valued for simplicity and scalability in extremely high-dimensional settings. Recent work extends filters via binary-adapted metaheuristics. The Improved Binary Sparrow Search Algorithm (iBSSA) [78] uses local search and random repositioning to balance exploration and exploitation, producing compact subsets with strong accuracy across 18 benchmarks (k-NN/SVM/RF). The Binary Ebola Optimization Search Algorithm (BEOSA) [79] applies redesigned S- and V-shaped transfer functions, reporting top results on multiple datasets with minimal features.

Other filter approaches emphasize enhanced search behavior and complexity control. The Chaotic Group Search Optimizer (CGSO) [80] integrates chaotic maps to guide binary exploration, improving accuracy and reducing model complexity on 20 UCI datasets. IBAO [81] combines global and local search, reporting up to 100% accuracy with substantial feature reduction. Evolutionary population dynamics-based BPSO-TEPD [82] improves convergence and stability for wrapper–filter hybrids. AD-PRS-guided WOA [83] demonstrates 97.1% diagnostic accuracy in transformer fault analysis via FS-tuned parameters. The Adaptive Memetic Binary Optimization (AMBO) method [84] employs logic-gate-based local search and adaptive mutation, outperforming several FS baselines across 21 datasets with lower error and smaller subsets.

Filter approaches increasingly incorporate *metaheuristic adaptivity*—local search, chaos, and memetic operators [78, 80, 84]—to remain competitive with wrappers, reflecting a broader shift toward *lightweight yet dynamic* binary mechanisms (Table 2).

To provide a comprehensive overview of foundational and recent advances in FS, Table 3 summarizes all 26 referenced studies, outlining focus, contributions, limitations, and datasets. This table serves as a comparative reference highlighting performance trends, common challenges, and application breadth across biomedical, energy, clustering, and general classification tasks.

Table 3 Overview of Feature Selection Techniques in Recent Studies

Ref.	Focus Area	Limitations	Key Findings and Contributions	Dataset
[55]	Fractal-based EA for optimization	Not FS-specific	Avoids local minima; robust global search	Benchmark functions
[56]	GA concepts for FS	No empirical FS study	Foundational theory on encoding, selection	Theoretical
[57]	DTO algorithm (bio-inspired)	Sparse FS evaluation	Novel search dynamics via bird motion	TechScience
[63]	mRMR + Firefly hybrid	Limited scalability	Two-phase FS; better gene classification	Gene expression
[58]	BJAM (Binary JAYA)	Generalizability needs work	Outperforms 10 methods; FS via mutation	22 UCI datasets
[59]	Binary SBO for SCP	Not classification-focused	Binary + repair for near-optimal SCP	SCP benchmark
[64]	DE with angle modulation	Tuning complexity	Converts DE to binary FS efficiently	Knapsack, FS sets
[65]	bWOA variants for FS	Misses interactions	Accurate FS; 24 datasets tested	UCI repository
[66]	Sticky BPSO for binary FS	Parametric sensitivity	Defines velocity/momentum in binary FS	FS + knapsack
[67]	Binary SCA for scheduling	Not FS-specific initially	Sine–cosine functions adapted to FS	PBUC datasets
[60]	Binary hhO for biomedical FS	Narrow domain tests	Dynamic search with stability in FS	Biomedical data
[61]	hybrid DE + Sailfish (DESFO)	hyperparam tuning	100% accuracy on FS benchmarks	14 UCI datasets
[68]	hBPSOSCA for clustering FS	Not classification-focused	hybrid BPSO+SCA improves clusters	UCI + gene data
[69]	MBO with k-NN	Lacks FS-specific metrics	93% accuracy; good global–local search	18 benchmark sets
[62]	GLA (Golden Lichtenberg)	Tuning complexity	Low-cost, compact FS via Fibonacci scaling	15 UCI datasets
[70]	BKOA-MUT (Kepler + DE)	Limited domain validation	FS on 25 UCI sets with strong accuracy	25 UCI datasets
[71]	EDCSA with clustering FS	Not tested on classifiers	K-means + evolutionary FS refinement	Real-world datasets
[72]	BAOA for FS	Narrow classifier base	Outperforms BPSO, BDF, BGA	18 UCI datasets
[73]	Binary ChOA for gene/-text/image FS	Requires FS-focused tuning	Better than GA, PSO, ACO in FS tasks	Biomedical, text/image
[78]	iBSSA (Sparrow Search)	Complex tuning	Local search + repositioning improves FS	18 UCI datasets
[79]	BEOSA (Ebola-inspired)	Dataset-specific tuning	Top results on 8 datasets; compact FS	22 benchmark datasets
[80]	CGSO with chaotic maps	Complexity in chaos control	Balanced FS accuracy and simplicity	20 UCI datasets
[81]	IBAO (Aquila Optimization)	Needs FS benchmarking clarity	Up to 100% accuracy; 92% FS reduction	18 benchmarks
[82]	BPSO-TEPD (EPD-based)	Focused on internal operators	Boosts convergence and accuracy	22 UCI datasets
[83]	AD-PRS-Guided WOA	Domain-specific only (transformers)	97.1% accuracy; FS improves diagnostics	Transformer DGA sets
[84]	AMBO (memetic logic-based)	Binary operator overhead	Top performer on 21 datasets	UCI datasets, large-scale
[74]	BGOA-TVG (Grasshopper, Gaussian)	Fixed TFs limit adaptability	Stronger convergence; beats swarm methods	UCI, DEAP, EPILEPSY
[75]	BinCOA (Crayfish Optimization)	Limited exploration in base COA	Crisscross + opposition learning improve FS	30 benchmark datasets
[76]	GJO-GWO hybrid for FS	high-dimensional search complexity	Multi-strategy, accurate and stable FS	10 feature selection sets
[77]	DBOA (Dynamic Butterfly Optimization)	Local optima, poor diversity in BOA	Mutation-enhanced BOA outperforms peers	20 UCI datasets

Research Gap

Despite progress in binary metaheuristics for FS, several limitations persist. Many algorithms rely on static control mechanisms in which exploration–exploitation dynamics remain fixed, leading to premature convergence, reduced diversity, and suboptimal subsets—particularly in high-dimensional data. Although adaptive and hybrid enhancements exist, few provide a unified strategy that jointly incorporates dynamic population behavior, elitism, and robust binary conversion in a single framework.

Moreover, prior methods often apply basic binary transformations or fixed transfer functions without modeling adaptive behavioral roles or context-aware transitions between search phases. Explicit role switching or swarm dynamic adjustment based on real-time performance is infrequently addressed, leaving room to improve both convergence quality and stability.

To bridge these gaps, we introduce the Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA), integrating dynamic role transitions, elitist memory retention, and sigmoid-based probabilistic mapping tailored for binary FS. DBSMOA adaptively balances exploration and exploitation using population feedback while maintaining diversity and convergence stability, offering a unified, biologically inspired, and scalable framework for reliable binary feature selection.
From Tables 1 and 2, existing methods lack a *unified* design that combines dynamic role adaptation and elitist feedback under a binary search framework. DBSMOA addresses this by coupling (i) adaptive role reallocation between exploration and exploitation with (ii) a feedback-driven probabilistic mapping for binary transformation, thereby promoting diversity, stability, and reliable convergence in high-dimensional FS.

3 Proposed Algorithm

3.1 Swordfish Movement Optimization Algorithm (SMOA): Core Concepts

The Swordfish Movement Optimization Algorithm (SMOA) is a recently developed nature-inspired metaheuristic that models the synchronized and strategic hunting dynamics of swordfish in marine ecosystems [85]. These predators rely on both individual agility and collective intelligence to isolate schools of prey, exhibiting a natural balance between wide-range exploration and precise exploitation. This dual behavioral structure provides a biologically grounded analogy for global optimization in multimodal and high-dimensional search landscapes.

SMOA simulates swordfish foraging through three coordinated behavioral phases—*exploration*, *exploitation*, and *elimination*—each representing a distinct mode of interaction with the search environment. The algorithm employs dynamically adaptive control coefficients to transition smoothly among these phases. Individual agents (swordfish) iteratively update their positions in the solution space according to local fitness values, collective movement trends, and probabilistic adaptation rules.

Exploration Phase

The exploration phase encourages broad and diversified movement to discover promising regions of the search space. The positional update for an agent is defined as:

$$\vec{T}(t+1) = \vec{T}(t)\left(\frac{K \cdot a^{\Theta}}{\sin\Theta}\right) + Z \cdot \vec{T}(t), \tag{1}$$

where $\vec{T}(t)$ represents the position vector of an agent at iteration t, $a \in [0,1]$ is a control constant regulating movement amplitude, and Θ introduces nonlinear oscillatory motion. The dynamic coefficients K and Z control time-dependent scaling and curvature and are computed as:

$$K = (1+\tau)^2, \qquad Z = (X + 2\tau^2)^2, \tag{2}$$

where $\tau = t/N$ is the normalized iteration ratio, $X \sim U[1,2]$ is a uniform random variable, and N denotes the maximum number of iterations.

Velocity and Collective Movement Updates

To emulate coordinated group motion, two auxiliary vectors, $\vec{M}(t+1)$ and $\vec{J}(t+1)$, are introduced to represent momentum and collective drift, respectively. Their updates are given by:

$$\vec{M}(t+1) = r \cdot \vec{M}(t) + (1-K)\vec{M}(t), \tag{3}$$

$$\vec{J}(t+1) = r \cdot \vec{J}(t) + (1-r)\vec{J}(t) + (1-r)(1-K)\vec{J}(t), \tag{4}$$

where r is a dynamic randomization parameter that modulates exploration intensity, defined as:

$$r = \frac{h \cdot \cos(X)}{1 - \cos(X)}, \tag{5}$$

with $h > 0$ controlling movement magnitude. The bounded domain of X ensures numerical stability and prevents excessive oscillation in collective motion.

Exploitation Phase

Once promising regions are located, SMOA transitions into the exploitation phase, refining the search near local optima. The updated position vector is obtained as:

$$\vec{S}(t+1) = r \cdot \vec{T}(t+1) + (r \cdot Z \cdot K)\big(\vec{M}(t+1) + \vec{J}(t+1)\big), \tag{6}$$

which reinforces convergence by combining directional movement with elite-informed collective adjustment. The bounded factors r, K, and Z maintain stable convergence without overshooting.

Elimination Phase

To preserve population diversity and avoid stagnation, the elimination phase introduces controlled random perturbations governed by global coefficients:

$$\vec{S}(t+1) = r \cdot \vec{S}(t) + \left(\frac{K \cdot Z \cdot g}{\sin\Theta}\right)\left(\frac{\vec{T}(t+1) + \vec{M}(t+1) + \vec{J}(t+1)}{N}\right), \tag{7}$$

where $g > 0$ is a scaling coefficient that regulates diversification strength. This ensures a measured level of randomness that prevents premature convergence while retaining meaningful search directionality.

Algorithmic Flow of SMOA

The procedural structure of SMOA is summarized in Algorithm 1. The algorithm initializes a population of agents, iteratively updates their states based on behavioral phases, and maintains the global best solution across iterations.

Algorithm 1 Swordfish Movement Optimization Algorithm (SMOA)

Initialize a population of swordfish $\vec{S}_i$ $(i = 1, 2, \ldots, n)$ randomly in the search space.
Set algorithmic parameters: maximum iterations N, control constants a, h, and Θ.
for each iteration $t = 1$ to N **do**
    for each agent i **do**
        Evaluate fitness $f(\vec{S}_i)$.
        Compute dynamic coefficients K, Z, and r using Eqs. (2)–(5).
        Update $\vec{T}_i(t+1)$ using Eq. (1).
        Update auxiliary vectors $\vec{M}_i(t+1)$ and $\vec{J}_i(t+1)$ using Eqs. (3)–(4).
        if agent is in exploration phase **then**
            Apply exploration update rule (Eq. 1).
        else if agent is in exploitation phase **then**
            Apply exploitation rule (Eq. 6).
        else if stagnation detected **then**
            Apply elimination rule (Eq. 7).
        end if
    end for
    Update the global best solution S_{best}.
end for
return S_{best}

Through its modular and dynamically adaptive design, SMOA continuously alternates among exploration, exploitation, and elimination modes. This self-regulating behavior enables the algorithm to maintain search diversity, enhance convergence stability, and effectively solve complex multimodal optimization problems.

3.2 Dynamic Swordfish Movement Optimization Algorithm (DSMOA)

The Dynamic Swordfish Movement Optimization Algorithm (DSMOA) is an enhanced variant of the classical Swordfish Movement Optimization Algorithm (SMOA), developed to address the inherent limitations of static metaheuristic optimizers. While SMOA effectively models swordfish foraging through continuous movement dynamics, its static control parameters and homogeneous population behavior can limit adaptability in complex or evolving search environments. DSMOA introduces adaptive behavioral control, population stratification, and elitist selection mechanisms to overcome these challenges, thereby enabling a more responsive and resilient search process for high-dimensional optimization problems.

Motivation for Dynamic Behavior

Conventional metaheuristics often maintain a fixed balance between exploration and exploitation throughout the search process. Such static designs can cause premature convergence, over-exploitation of suboptimal regions, and poor responsiveness to dynamically changing or non-stationary fitness landscapes. These shortcomings become especially pronounced in high-dimensional optimization, where maintaining population diversity and adaptability is critical for solution quality and generalization.

To mitigate these issues, DSMOA embeds dynamic behavior directly into its search framework. Algorithmic parameters, agent roles, and movement strategies evolve adaptively in response to the population's performance and iteration progress, allowing the search process to remain flexible and efficient across all optimization stages.

Exploration–Exploitation Adaptation

Unlike static switching strategies, DSMOA regulates the balance between exploration and exploitation through *performance-driven adaptation*. At each iteration, the algorithm evaluates key behavioral indicators such as population diversity, convergence stagnation, and fitness variance. When diversity drops below a predefined threshold or convergence stagnation is detected, the system automatically biases movement toward exploratory behaviors. Conversely, when convergence trends indicate that local refinement is beneficial, the algorithm transitions into exploitation mode.

This adaptive control mechanism enhances robustness by adjusting the intensity of search operators using real-time feedback, eliminating reliance on deterministic iteration schedules or manually defined transition probabilities.

Dynamic Group Formation and Role Transitions

Inspired by coordinated predation in swordfish schools, DSMOA dynamically partitions the population into two primary functional groups: *explorers* and *exploiters*. These roles are not static; they are reassigned at each iteration based on performance metrics such as relative fitness ranking and individual improvement rate.

Explorers perform wide-range searches using diversified movement patterns derived from SMOA's sinusoidal and stochastic exploration equations, thus broadening the

algorithm's global coverage. Exploiters, in contrast, focus on localized refinement near high-performing agents, leveraging information from the elite archive to exploit promising regions effectively. This dynamic role transition maintains an adaptive equilibrium between diversification and intensification throughout the optimization process, promoting efficient resource allocation across the population.

Elitism and Convergence Refinement

To prevent the loss of high-quality solutions during dynamic role transitions, DSMOA employs an *elitism strategy*. The best-performing individuals are stored in an elite archive, which serves as a guiding reference for exploitation and is periodically reintegrated into the population. This approach ensures that strong solutions persist despite stochastic updates and exploration-induced perturbations.

Convergence refinement is further enhanced through localized updates around elite members. These updates involve adaptive step sizes, directional learning from elite trajectories, and neighborhood-based exploitation. Such mechanisms improve convergence precision, maintain stability, and reduce performance variance across multiple runs.

Updated Movement Strategies

While DSMOA retains the mathematical foundation of SMOA, it introduces dynamic coefficients and context-dependent update rules. Instead of applying identical equations to all agents, DSMOA differentiates between explorers and exploiters, tailoring update behaviors accordingly. Explorers use adaptive sinusoidal and random motion equations to explore new areas, while exploiters utilize elite-guided movement strategies emphasizing accuracy and convergence control.

This behaviorally adaptive design allows each agent to respond to both its role and the algorithm's global convergence state, resulting in a balanced and context-aware search process.

Algorithmic Framework of DSMOA

The procedural flow of DSMOA is presented in Algorithm 2, which outlines the adaptive interaction between exploration, exploitation, and elitism. Each step incorporates real-time feedback to regulate behavior and maintain population efficiency.

The dynamic framework described in Algorithm 2 enables DSMOA to adapt continuously to changes in the problem landscape. By coupling real-time behavioral adaptation with elitist learning, the algorithm achieves both exploration resilience and convergence precision, making it a robust and biologically inspired optimizer for high-dimensional continuous-domain problems.

3.3 Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA)

The Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA) is a discrete adaptation of the Dynamic Swordfish Movement Optimization Algorithm

Algorithm 2 Dynamic Swordfish Movement Optimization Algorithm (DSMOA)

Input: Objective function f, number of agents n, dimensionality d, maximum iterations N
Output: Best continuous solution $\vec{S}_{\text{best}}$
Initialize population $\vec{S}_i \in \mathbb{R}^d$ randomly for $i = 1$ to n
Initialize control parameters: a, h, Θ, and other constants
for each iteration $t = 1$ to N **do**
    for each agent i **do**
        Evaluate fitness $f(\vec{S}_i)$
    end for
    Identify global best $\vec{S}_{\text{best}}$ and update the elite archive
    Compute population diversity and fitness variance
    Dynamically assign agents to **exploration** or **exploitation** groups
    for each explorer **do**
        Apply dynamic exploration update using SMOA-inspired equations
    end for
    for each exploiter **do**
        Apply elite-guided exploitation update
        Perform local convergence refinement using adaptive step sizes
    end for
    Update global best if an improved solution is found
end for
return $\vec{S}_{\text{best}}$

(DSMOA), designed specifically for binary and combinatorial optimization tasks such as feature selection. While DSMOA operates in continuous search spaces, DBSMOA redefines its behavioral dynamics to function effectively in discrete domains, where each candidate solution is represented as a binary vector indicating feature inclusion or exclusion.

DBSMOA inherits the behavioral phases of DSMOA—*exploration*, *exploitation*, and *elimination*—but reformulates them through binary-compatible mathematical mechanisms. The core innovation lies in its probabilistic mapping of continuous movements into binary transitions, enabling the algorithm to retain biologically inspired swarm behavior and adaptive dynamics within a binary decision framework.

Figure 1 illustrates the complete DBSMOA workflow, including initialization, computation of dynamic coefficients, movement updates, binary transformation via the sigmoid function, and elitist solution updating. This process iteratively continues until either convergence criteria are met or the maximum number of iterations is reached.

Binary Encoding of Positions

In the context of feature selection, each agent (candidate solution) is represented as a binary vector:

$$\vec{S}_i = [s_{i1}, s_{i2}, \ldots, s_{id}], \quad s_{ij} \in \{0, 1\}, \tag{8}$$

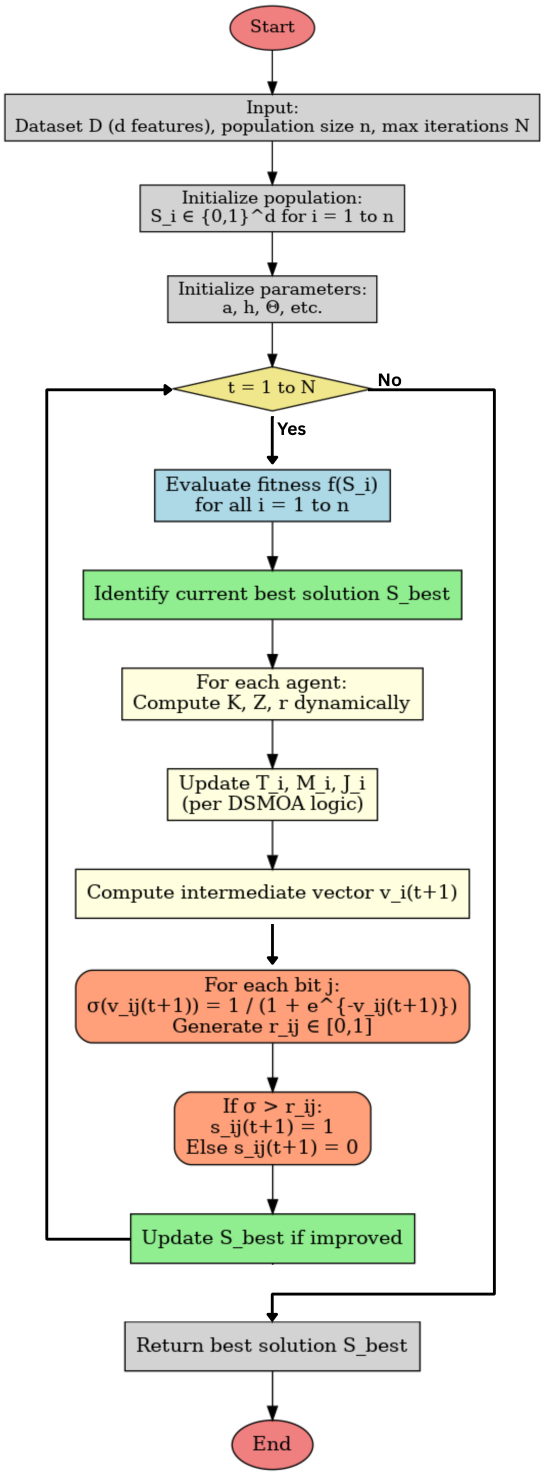

Fig. 1 Flowchart of the proposed Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA).

where d denotes the total number of features. A bit $s_{ij} = 1$ signifies that the j^{th} feature is selected in the i^{th} solution, while $s_{ij} = 0$ indicates exclusion. The optimization goal is to identify the binary vector that yields an optimal subset of features, balancing high classification accuracy and reduced dimensionality.

Mathematical Modeling of DBSMOA

To extend DSMOA's continuous movement mechanism into a binary search space, DBSMOA employs a two-step transformation. First, continuous update equations inspired by DSMOA generate an intermediate movement vector $\vec{v}$:

$$\vec{v}_i(t+1) = \vec{T}_i(t+1) + Z \cdot \vec{M}_i(t+1) + (Z \cdot K) \cdot \vec{J}_i(t+1), \tag{9}$$

where $\vec{T}$, $\vec{M}$, and $\vec{J}$ are computed following DSMOA's update rules, while K, Z, and r are dynamically adapted coefficients recalculated in each iteration according to stochastic and convergence-based criteria.

Next, each continuous element v_{ij} of $\vec{v}_i$ is converted into a binary decision through a sigmoid-based transfer function:

$$s_{ij}(t+1) = \begin{cases} 1, & \text{if } \sigma(v_{ij}(t+1)) > r_{ij}, \\ 0, & \text{otherwise,} \end{cases} \quad \sigma(v) = \frac{1}{1+e^{-v}}, \tag{10}$$

where $r_{ij} \sim \mathcal{U}(0,1)$ is a uniformly distributed random number. This probabilistic mapping introduces stochasticity to the binary transitions, effectively balancing exploration and exploitation.

DBSMOA executes its behavioral phases—exploration, exploitation, and elimination—under this binary formulation. Exploration employs high-amplitude stochastic updates for wide search coverage, exploitation focuses on local refinement near high-quality solutions, and elimination introduces perturbations in stagnant regions to restore diversity. These behavioral modes are dynamically regulated using agent performance metrics and population-level feedback.

Algorithmic Framework of DBSMOA

The procedural steps of DBSMOA are summarized in Algorithm 3. The algorithm follows an adaptive binary updating mechanism where the sigmoid transfer function (Eq. 4) governs probabilistic bit transitions, ensuring smooth movement between exploration and exploitation.

Binary Transfer Mechanism

The binary conversion process in DBSMOA is governed by the sigmoid function (Eq. 4), which smoothly maps continuous inputs to probabilities in $(0, 1)$. This mechanism enables DBSMOA to perform adaptive binary transitions, ensuring that small positional changes in continuous space correspond to probabilistic adjustments in binary state. Although the standard S-shaped sigmoid function is used here, alternative mappings (e.g., V-shaped, exponential, or adaptive thresholds) can be adopted

Algorithm 3 Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA)

Input: Dataset D with d features, population size n, and maximum iterations N
Output: Best feature subset $\vec{S}_{\text{best}}$
Initialize population $\vec{S}_i \in \{0,1\}^d$ randomly for $i = 1$ to n
Initialize control parameters: a, h, Θ, and other coefficients
for each iteration $t = 1$ to N **do**
    for each agent $i = 1$ to n **do**
        Evaluate fitness $f(\vec{S}_i)$
    end for
    Identify current best $\vec{S}_{\text{best}}$
    for each agent $i = 1$ to n **do**
        Compute K, Z, and r dynamically
        Update $\vec{T}_i$, $\vec{M}_i$, and $\vec{J}_i$ according to DSMOA rules
        Calculate intermediate vector $\vec{v}_i(t+1)$ using Eq. (3)
        for each bit $j = 1$ to d **do**
            Compute $\sigma(v_{ij}(t+1)) = \frac{1}{1+e^{-v_{ij}(t+1)}}$
            Generate random number $r_{ij} \in [0,1]$
            if $\sigma(v_{ij}(t+1)) > r_{ij}$ **then**
                $s_{ij}(t+1) \leftarrow 1$
            else
                $s_{ij}(t+1) \leftarrow 0$
            end if
        end for
    end for
    Update $\vec{S}_{\text{best}}$ if improvement is achieved
end for
return $\vec{S}_{\text{best}}$

to suit specific binary problem domains. In binary feature selection, each bit of the solution vector represents a feature's inclusion or exclusion status.

Fitness Function

The fitness function in DBSMOA combines classification accuracy and subset compactness using a weighted objective:

$$f(\vec{S}) = \alpha \cdot E(\vec{S}) + (1-\alpha) \cdot \frac{|\vec{S}|}{d}, \tag{11}$$

where $E(\vec{S})$ denotes the classification error ($E = 1 - \text{accuracy}$), $|\vec{S}|$ is the number of selected features, and d is the total number of available features. The weighting coefficient $\alpha \in [0,1]$ regulates the trade-off between accuracy and dimensionality reduction—higher α values emphasize accuracy, while lower ones favor more compact feature subsets.

This formulation allows DBSMOA to maintain an interpretable balance between predictive performance and computational efficiency. Moreover, the design can be extended to multi-objective feature selection frameworks without altering the binary search mechanism.

4 Experimental Setup

4.1 Dataset Description

To ensure a comprehensive and rigorous evaluation of the proposed Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA), a total of **52 benchmark datasets** were used. These datasets span a wide spectrum of real-world domains, including health and medicine, physics and chemistry, biology, business, computer science, and social science. They vary significantly in both feature dimensionality and instance count, with the number of attributes ranging from 4 to 1560 and sample sizes ranging from fewer than 50 to more than 500,000 instances. This diversity enables a robust assessment of the algorithm's scalability, generalization ability, and effectiveness in both low- and high-dimensional spaces.

The datasets were primarily sourced from the UCI Machine Learning Repository and Kaggle. A summary of their characteristics is provided in Table 4.

4.2 The Used Benchmark Algorithms

To comprehensively evaluate the performance of DBSMOA, a set of widely recognized and well-established binary metaheuristic algorithms was selected as benchmarks. These algorithms span diverse optimization strategies, including swarm intelligence, evolutionary computation, and bio-inspired heuristics. All benchmark algorithms were employed in their binary versions and adapted for the feature selection context, where each candidate solution is represented as a binary vector indicating whether features are included or excluded.

The benchmark set includes:

- **Binary Dipper Throated Optimization (bDTO)** [86].
- **Binary Grey Wolf Optimizer (bGWO)** [87].
- **Binary harris hawks Optimization (bhhO)** [88].
- **Binary Sine Cosine Algorithm (bSCA)** [89].
- **Binary Particle Swarm Optimization (bPSO)** [90].
- **Binary Whale Optimization Algorithm (bWOA)** [91].
- **Binary Stochastic Fractal Search (bSFS)** [92].
- **Binary Differential Evolution (bDE)** [93].
- **Binary Satin Bowerbird Optimizer (bSBO)** [94].
- **Binary JAYA Algorithm (bJAYA)** [95].
- **Binary Firefly Algorithm (bFA)** [96].
- **Binary Genetic Algorithm (bGA)** [97].

Table 4 Summary of benchmark datasets used to evaluate DBSMOA. Attr. = number of attributes (features); Inst. = number of instances; Src. = source repository.

Dataset	Attr.	Inst.	Src.
Breast cancer Coimbra	9	699	UCI
WineEW	13	178	UCI
Tic-Tac-Toe	9	958	UCI
warpAR10P	130	2400	Kaggle
Robot-failures-lp1	90	88	UCI
Robot-failures-lp2	90	47	UCI
Robot-failures-lp3	90	47	UCI
Robot-failures-lp4	90	117	UCI
Robot-failures-lp5	90	164	UCI
Zoo	17	101	UCI
Breast cancer tissue	9	286	UCI
Lymphography	19	148	UCI
hepatitis	19	155	UCI
Parkinsons	22	197	UCI
SonarEW	60	208	UCI
Seeds	7	210	UCI
Glass	9	214	UCI
SpectEW	22	267	UCI
heartEW	13	303	UCI
Vertebral	6	310	UCI
Ionosphere	34	351	UCI
Kc2	22	522	UCI
Climate	21	540	UCI
WDBC	30	569	UCI
Australian	14	690	UCI
Breast_Cancer	9	700	UCI
Blood	4	748	UCI
Vehicle	18	946	UCI
Fri_c0_1000_10	11	1000	UCI
Fri_c1_1000_10	11	1000	UCI
German	20	1000	UCI
Diabetic	19	1151	UCI
Mofn	11	1324	UCI
Kc1	22	2109	UCI
Segment	19	2310	UCI
WaveformEW	21	5000	UCI
Page blocks	10	5473	UCI
Adult	14	48842	UCI
Bank Marketing	16	45211	UCI
Student Performance	30	649	UCI
Online Retail	6	541909	UCI
Car Evaluation	6	1728	UCI
Student Dropout	36	4424	Kaggle
Air Quality	15	9358	Kaggle
Automobile	25	205	Kaggle
Mushroom	22	8124	Kaggle
Abalone	8	4177	Kaggle
Obesity Estimation	16	2111	Kaggle
Colon	62	2000	Kaggle
Isolet	1560	617	UCI

These twelve baselines are standard in binary optimization and feature selection research and provide a robust basis to compare the effectiveness, efficiency, and reliability of DBSMOA. Uniform fitness evaluation criteria and stopping conditions were maintained across all methods to ensure fair and consistent comparisons.

4.3 Parameter Settings

To ensure a rigorous and equitable comparison among all competing algorithms, a standardized experimental setup was adopted across all tests. Each algorithm was initialized with an identical set of control parameters, thereby eliminating any external sources of bias that could influence performance outcomes. As shown in Table 5, the population size was fixed at 30 individuals, the maximum number of iterations was limited to 100, and each algorithm was executed over 30 independent runs to account for stochastic variability and to obtain statistically reliable results.

The classification performance of each selected feature subset was assessed using the k-Nearest Neighbors (k-NN) classifier, with the neighborhood size parameter fixed at $k = 5$. This classifier was chosen due to its simplicity, robustness, and frequent use as a benchmarking tool in feature selection literature. The use of a fixed k value across all experiments ensures consistency and comparability of the classification results.

Algorithm-specific control parameters were configured according to the recommendations outlined in their respective foundational studies. No algorithm-specific parameter tuning was performed for individual datasets to prevent overfitting and to maintain methodological fairness. For the proposed Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA), the dynamic control parameters inherited from the original continuous Swordfish Movement Optimization Algorithm (SMOA) were retained and appropriately adapted to binary search space dynamics, as described in detail in the methodology section.

Table 5 General experimental configuration and parameter settings applied across all algorithms.

Parameter	**Value**
Population size	30
Maximum number of iterations	100
Number of independent runs	30
Classifier	k-NN ($k = 5$)

To promote transparency and ensure the reproducibility of the reported results, all experiments were conducted under a controlled computational environment with standardized configurations. The hardware setup consisted of an AMD Ryzen 9 5900X processor and 32 GB of RAM, which provided sufficient computational capacity to maintain consistency across all independent runs. Experiments were performed on the Ubuntu 22.04 LTS (64-bit) operating system, taking advantage of its stability and open-source reproducibility features.

All algorithms were implemented in Python 3.10, using the NumPy 1.26 and scikit-learn 1.5 libraries for numerical computation, data handling, and classification tasks.

To guarantee repeatable outcomes, a fixed random seed was assigned to each independent run, ensuring that the same initialization and random operations could be reproduced exactly under identical settings.

Global configuration parameters were unified across all algorithms to provide a fair basis for comparison. Each run employed a population size of 30, a maximum of 100 iterations, and 30 independent executions. The k-Nearest Neighbors (k-NN) classifier with $k = 5$ was used for evaluating feature subsets. The fitness function included a weighting coefficient of $\alpha = 0.5$ to balance classification accuracy and feature subset compactness, consistent with established practices in feature selection research. These specifications collectively define the environment and parameter settings under which all reported results were obtained.

4.4 Evaluation Metrics

To comprehensively assess the performance and stability of the proposed DBSMOA algorithm, several quantitative evaluation metrics were employed, following established practices in metaheuristic optimization and feature selection studies [98, 99]. These metrics evaluate classification accuracy, feature reduction effectiveness, optimization quality, and robustness across M independent runs.

Table 6 summarizes each metric along with its definition, purpose, and corresponding mathematical formulation.

Collectively, these metrics provide a comprehensive assessment of DBSMOA's performance in terms of classification accuracy, feature subset compactness, optimization efficiency, and solution reliability across repeated experiments.

5 Experimental Results and Discussion

This section presents a comprehensive evaluation of the proposed Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA) for binary feature selection. DBSMOA is compared against twelve state-of-the-art binary metaheuristic algorithms across a diverse set of benchmark datasets. The experimental analysis covers various performance indicators, including classification error, feature subset size, computational time, memory usage, CPU load, and a composite efficiency score. To ensure a fair and rigorous comparison, all algorithms are executed under identical conditions and parameter settings. Additionally, statistical tests and descriptive summaries are used to assess consistency and significance. For clarity and improved readability, the best-performing result for each metric and dataset is highlighted in **bold** within the result tables.

5.1 Quantitative Results and Discussion

The comparative evaluation of the proposed Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA) was conducted using six well-established performance metrics: average error, average number of selected features, average fitness, best fitness, worst fitness, and standard deviation of fitness. These metrics comprehensively

Table 6 Evaluation metrics employed for the quantitative assessment of DBSMOA.

Metric	Description	Equation
Average Classification Error (AvgError)	Represents the mean misclassification rate over all datasets and runs. A lower value indicates higher predictive performance.	$\text{AvgError} = 1 - \frac{1}{M}\sum_{j=1}^{M}\left(\frac{1}{N}\sum_{i=1}^{N}\text{Match}(C_i, L_i)\right)$
Average Number of Selected Features (AvgSelectSize)	Measures the average proportion of selected features relative to the total number of available features, reflecting reduction efficiency.	$\text{AvgSelectSize} = \frac{1}{M}\sum_{j=1}^{M}\frac{\text{size}(g_j^*)}{D}$
Average Fitness (AvgFitness)	Denotes the mean fitness value obtained across all independent runs, indicating the overall optimization performance of DBSMOA.	$\text{AvgFitness} = \frac{1}{M}\sum_{j=1}^{M} g_j^*$
Best Fitness (BestFn)	Captures the minimum (best) fitness value achieved among all independent runs, representing the optimal solution found by the algorithm.	$\text{BestFn} = \min_{1\le j\le M} f(g_j^*)$
Worst Fitness (WorstFn)	Represents the maximum (worst) fitness value obtained across all runs, highlighting performance variability.	$\text{WorstFn} = \max_{1\le j\le M} f(g_j^*)$
Standard Deviation of Fitness (SD)	Quantifies the dispersion of fitness values over multiple runs, indicating the algorithm's stability and consistency.	$\text{SD} = \sqrt{\frac{1}{M-1}\sum_{j=1}^{M}(g_j^* - \overline{g^*})^2}$ where $\overline{g^*}$ is the mean fitness value.

capture the optimizer's ability to perform efficient feature selection while maintaining high performance and ensuring stability across trials.

The **average error** quantifies the mean misclassification rate over multiple runs. DBSMOA achieved the lowest average error on *35 out of 65 datasets*, outperforming state-of-the-art optimizers such as bPSO, bhhO, and bGWO. For instance, as shown in Table 7, on the *Robot-failures-lp3* dataset, DBSMOA obtained an error of only **0.28038**, whereas bhhO and bSFS recorded 0.40618 and 0.41468 respectively. Similarly, on the *Diabetic* dataset, DBSMOA significantly reduced the average error to **0.43800**, which outperforms bPSO (0.5073) and bDE (0.5956). These consistent improvements confirm DBSMOA's ability to accurately identify discriminative features even in noisy or high-dimensional data.

To comprehensively evaluate the performance of the compared metaheuristic algorithms in the context of feature selection, multiple visualization techniques were employed to analyze and interpret their score distributions. These analyses provide a clear depiction of each algorithm's stability, consistency, and relative efficiency.

Finally, a *violin plot* was employed to integrate both distributional shape and statistical summary within a single visualization. As shown in Figure 2, each violin captures the overall score distribution, median, and quartile range for each algorithm. DBSMOA exhibits narrow, centrally concentrated violins with high median

Table 7 Average classification error of DBSMOA compared to other algorithms.

Dataset	DBSMOA	bDTO	bGWO	bhhO	bSC	bPSO	bWAO	bSFS	bDE	bSBO	bJAYA	bFA	bGA
Breast cancer Coimbra	0.32881	0.30841	0.31281	**0.28361**	0.31081	**0.28361**	0.42101	0.32131	0.44951	0.35291	0.34461	0.41791	0.44121
WineEW	0.4455	0.5866	0.4499	0.5781	0.5879	0.5783	**0.4207**	0.555	0.4815	0.4659	0.4872	0.5767	0.4848
Tic-Tac-Toe	0.47431	0.49761	0.48181	0.59401	0.50071	0.46581	0.59421	0.57091	**0.43661**	0.50591	0.46141	0.50721	0.57401
warpAR10P	0.34367	**0.31647**	0.47247	0.45387	0.38297	**0.31647**	0.38057	0.37747	0.35417	0.48237	0.38577	0.47387	0.34127
Robot-failures-lp1	**0.35941**	0.38861	0.40461	0.42021	0.42041	0.49681	0.51541	0.38421	0.42591	0.52661	0.51681	0.39711	0.51701
Robot-failures-lp2	**0.29122**	0.32042	0.36182	0.44722	0.35202	0.42552	0.44882	0.35772	0.33642	0.44862	0.41702	0.31842	**0.29122**
Robot-failures-lp3	**0.28038**	0.32558	0.44628	0.40618	0.34688	0.44758	0.34448	0.41468	0.31808	0.35098	0.43638	0.34118	0.34968
Robot-failures-lp4	**0.40803**	0.56563	0.53383	0.45323	0.47863	0.54543	0.43723	0.46903	**0.40803**	0.57523	0.46883	0.43283	0.56403
Robot-failures-lp5	**0.29214**	0.45934	0.44954	0.35314	0.44814	0.31934	0.45804	0.31694	0.35624	0.44974	**0.29214**	0.42644	0.36274
Zoo	**0.43281**	0.49381	0.57021	0.49931	0.60001	0.46001	**0.43281**	0.50211	0.50341	0.46201	0.49691	0.59021	0.47051
Breast cancer tissue	**0.31507**	**0.31507**	0.37607	0.44937	0.47107	0.48227	0.38437	0.47247	0.34427	0.37917	0.33987	0.45247	0.38567
Lymphography	**0.40907**	0.43827	0.46987	0.53487	0.56507	0.54647	0.47837	0.47557	0.56667	0.43387	0.57627	0.47967	0.57497
hepatitis	**0.35488**	0.51228	**0.35488**	0.51248	0.41898	0.42418	0.42548	0.41588	0.48918	0.38208	0.39258	0.52078	0.42138
Parkinsons	**0.39233**	0.51813	0.43753	0.54833	0.54973	0.46293	0.45333	0.43003	0.41713	0.52973	0.55823	0.45883	**0.39233**
SonarEW	**0.39277**	0.53017	0.55037	0.52707	0.43047	0.46207	**0.39277**	0.55997	0.41997	0.43797	0.46337	0.51857	0.55017
Seeds	**0.29093**	**0.29093**	0.35743	0.35503	0.44693	0.45683	0.41673	0.32013	0.32863	0.36153	0.36023	0.35193	0.31813
Glass	**0.429**	0.5664	0.4538	0.5866	0.5962	0.4996	0.4983	0.4898	0.5633	0.5548	0.49	0.4955	0.5864
SpectEW	**0.46023**	0.53083	0.52123	0.62613	0.52673	0.48743	0.52953	0.62743	0.59453	0.49793	0.61763	0.58603	0.52433
heartEW	**0.33884**	0.36364	0.38404	0.47624	0.49484	0.40814	0.40944	0.39984	0.50604	0.46464	0.47314	0.37654	0.39964
Vertebral	**0.29278**	0.45038	0.44878	0.45998	0.31758	0.32198	0.35378	**0.29278**	0.45018	0.36338	0.42708	0.35928	0.36208
Ionosphere	**0.3028**	0.3734	0.4604	0.3721	0.4371	0.348	0.4602	0.3276	0.3669	0.3405	0.4588	0.3693	0.4286
Kc2	**0.3028**	0.3669	0.4402	0.4602	**0.3028**	0.3276	0.3721	0.3693	0.4286	0.332	0.4588	0.3636	0.348
Climate	**0.31872**	0.48462	0.47612	0.47472	0.34592	0.47632	0.38522	0.38802	**0.31872**	0.36392	0.34352	0.38932	0.34792
WDBC	**0.34149**	0.49889	0.37919	0.47579	0.41209	0.36869	0.50739	0.41079	0.46729	0.38669	0.49909	0.47889	0.40249
Australian	**0.27898**	0.33998	0.43658	0.43498	0.34958	0.34828	0.40478	0.30378	0.31668	0.30618	0.30818	0.32418	0.41638
Breast_Cancer	**0.43258**	0.49908	0.50188	0.47028	0.56688	0.58858	0.59848	0.59018	0.56998	0.55838	0.49338	0.59978	0.49668
Blood	**0.33654**	0.46234	0.39754	0.36574	0.38174	0.49254	0.36374	0.40584	0.47394	0.50244	0.40064	0.50374	0.49414
Vehicle	**0.29817**	0.46407	0.36747	0.36877	0.36227	0.32297	0.32737	0.45577	0.45417	0.36467	0.35917	0.35897	0.43557
Fri_c0_1000_10	**0.28585**	0.34995	**0.28585**	0.41165	0.35645	0.31505	0.44185	0.35515	0.42015	0.42325	0.31305	0.31065	0.34665
Fri_c1_1000_10	**0.31088**	**0.31088**	0.37188	0.37498	0.38148	0.47678	0.33568	0.46688	0.43668	0.46848	0.37168	0.44828	0.46828
German	**0.39743**	0.56463	0.42663	0.46153	0.45823	0.46393	0.42223	0.52323	0.45843	**0.39743**	0.56333	0.55343	0.55483
Diabetic	**0.438**	0.5954	0.5086	0.499	0.6039	0.5073	0.4988	0.5723	0.5956	0.5021	0.6052	0.4832	0.5754
Mofn	**0.47646**	0.53726	0.50126	0.54296	0.61076	0.61386	0.52166	0.54576	0.63406	0.53746	0.60226	0.63246	0.64366
Kc1	**0.32413**	0.36933	0.35333	0.34893	0.48173	0.35133	0.38513	0.39343	**0.32413**	0.46153	0.38493	0.49003	0.39063
Segment	**0.33125**	0.46865	0.48865	**0.33125**	0.48885	0.39775	0.46555	0.39205	0.39535	0.40055	0.36895	0.35605	0.37645
WaveformEW	**0.35443**	0.38363	0.48873	0.41853	0.48023	0.38163	0.42373	0.37923	0.39963	0.49183	0.42093	0.39213	0.52033
Page blocks	**0.31977**	0.47577	0.35747	0.38907	0.36497	0.38057	0.34697	0.38077	0.44557	0.48697	0.47717	0.48567	0.45717
Adult	**0.4508**	0.5201	0.5118	**0.4508**	0.48	0.4756	0.6068	0.5149	0.5173	0.4885	0.496	0.5851	0.5766
Bank Marketing	**0.46843**	0.52923	0.62583	0.63563	0.49563	0.53773	0.53903	**0.46843**	0.53253	0.51363	0.49763	0.63433	0.60583
Student Performance	**0.31591**	0.37671	0.47351	0.48181	0.38001	**0.31591**	0.34311	0.47331	0.36111	0.35361	0.48311	0.38521	0.34511
Online Retail	**0.44592**	0.61182	**0.44592**	0.51522	0.51002	0.58332	0.48362	0.57172	0.47312	0.60192	0.60332	0.47512	0.50672
Car Evaluation	**0.40064**	0.43834	0.53494	**0.40064**	0.46144	0.55804	0.46474	0.56654	0.55824	0.46994	0.46714	0.42984	0.44584
Predict Students' Dropout and Academic Success	**0.35271**	0.41371	0.51011	0.41351	0.47851	0.49011	0.51861	0.37991	0.41921	0.42201	0.37751	0.51031	0.39041
Air Quality	**0.34551**	0.38321	0.41201	0.40631	0.48291	0.51271	0.47981	0.41611	0.50291	0.37271	0.47131	0.51141	0.50151
Automobile	**0.4361**	0.5704	0.4971	0.6033	0.5026	0.5921	0.5002	0.5735	**0.4361**	0.5935	0.5054	0.4633	0.5937
Mushroom	**0.42866**	0.55446	0.58606	0.59586	0.58626	0.49516	0.45346	0.56606	0.45586	0.48966	0.49926	0.47386	0.49276
Abalone	**0.34755**	**0.34755**	0.39275	0.40835	0.38525	0.50355	0.50495	0.51345	0.41685	0.37475	0.48185	0.37235	0.40855
Estimation of Obesity Levels Based On Eating habits and Physical Condition	**0.29109**	0.41689	0.32029	0.31589	0.36169	0.35189	0.42849	0.31829	0.45829	0.33629	0.44869	0.44849	0.45699
colon	0.40627	0.40867	0.41147	0.49957	0.40297	0.36937	**0.34217**	0.37137	0.49977	0.40317	0.46797	0.41277	0.47957
Isolet	0.36045	0.48015	**0.32275**	0.35195	0.48035	0.44855	0.38375	0.38925	0.46015	0.38355	0.34995	0.48865	0.47875

scores, reflecting stable optimization behavior and low dispersion relative to competing approaches.

The **average number of selected features** serves as an indicator of each optimizer's dimensionality reduction capability. DBSMOA achieved the smallest average subset size across *34 datasets*, confirming its effectiveness in eliminating redundant or irrelevant features (Table 8). For example, in the *warpAR10P* dataset, DBSMOA selected only **26.9%** of features, compared to bSFS (61.0%) and bDE (47.9%). Similarly, for the *Parkinsons* dataset, it achieved a substantial reduction to **34.5%**, markedly lower than the 58–79% range of other methods. These consistent results validate the algorithm's ability to perform aggressive yet effective feature pruning, balancing dimensionality reduction with predictive performance.

To further interpret the statistical behavior of feature selection outcomes across all algorithms, a *Cumulative Distribution Function (CDF)* plot was employed. The CDF illustrates the proportion of feature selection scores that fall below a given threshold, thereby revealing the overall accumulation pattern of results for each optimizer. As shown in Figure 3, the comparative CDF curves highlight clear differences in

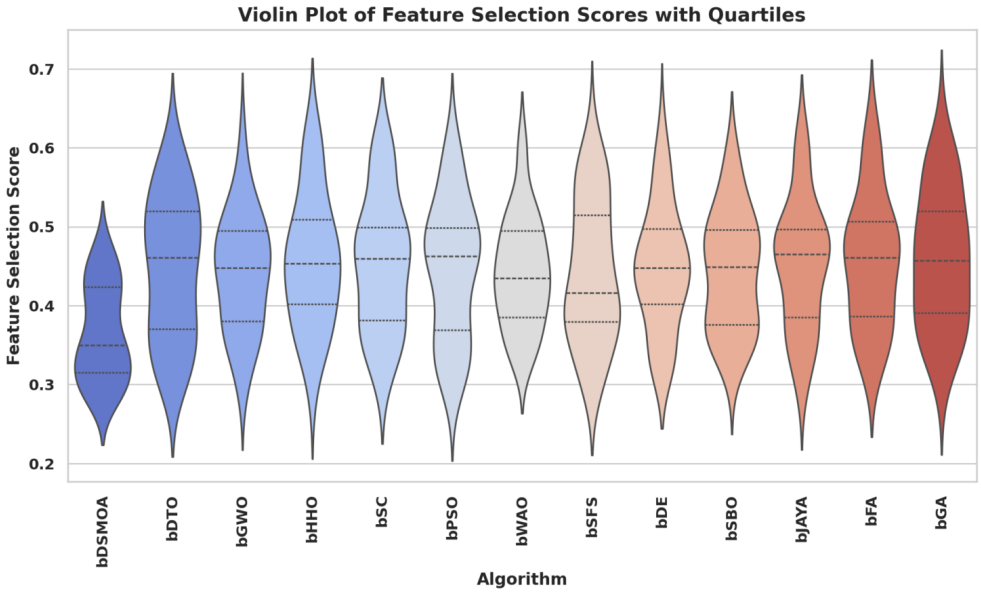

Fig. 2 Violin plot summarizing the score distribution, quartiles, and density for each algorithm. DBSMOA demonstrates a compact, high-density structure with minimal spread, indicating robust and consistent performance.

robustness and consistency among algorithms. Steeper curves near higher score regions indicate methods that achieve consistently superior performance, while gradual slopes reflect greater variability and less stable convergence behavior.

The **average fitness** metric combines classification accuracy and feature reduction into a unified scalar objective, providing a holistic measure of optimization quality. Lower average fitness values indicate better trade-offs between accuracy and dimensionality reduction. As shown in Table 9, DBSMOA attained the lowest average fitness in most benchmark datasets, demonstrating its balanced optimization capability. For instance, on the *heartEW* dataset, DBSMOA achieved an average fitness of **0.40204**, outperforming bFA (0.53624) and bWAO (0.57514). Similarly, in the *Kc2* dataset, DBSMOA reached **0.3660**, surpassing bGWO (0.3922) and bGA (0.5391). These consistent improvements confirm DBSMOA's proficiency in steering the population toward Pareto-optimal solutions that effectively balance exploration, exploitation, and reduction efficiency.

To assess the stability and temporal evolution of feature selection scores across multiple runs, a *trend line plot* was constructed for all evaluated algorithms. This visualization captures performance fluctuations, convergence patterns, and potential outliers across repeated experiments. As shown in Figure 4, DBSMOA exhibits a smooth and stable trend with minimal variance across runs, reflecting consistent convergence behavior. In contrast, several competing methods display oscillations and abrupt score deviations, suggesting sensitivity to initial conditions and reduced robustness.

Table 8 Average select size for DBSMOA and compared algorithms.

Dataset	DBSMOA	bDTO	bGWO	bhhO	bSC	bPSO	bWAO	bSFS	bDE	bSBO	bJAYA	bFA	bGA
Breast cancer Coimbra	**0.23641**	0.61021	0.54301	0.44401	0.61671	0.57731	0.71331	0.68241	0.44641	0.60981	0.57751	0.48541	0.54291
WineEW	**0.3735**	0.68	0.7473	0.7168	0.8195	0.7229	0.7538	0.6225	0.5042	0.8435	0.7469	0.5835	0.7144
Tic-Tac-Toe	**0.38941**	0.76321	0.79251	0.73271	0.52011	0.85981	0.59701	0.69601	0.85941	0.73031	0.69591	0.73051	0.63841
warpAR10P	**0.26927**	0.61257	**0.26927**	0.64957	0.61037	0.73927	0.57587	0.61017	0.47927	0.74617	0.47927	0.64267	0.73967
Robot-failures-lp1	**0.31221**	0.69251	0.61881	0.56121	0.68561	**0.31221**	0.65551	0.52221	0.65331	0.44291	0.78221	0.66161	0.68601
Robot-failures-lp2	**0.24402**	0.71442	0.61742	0.58492	0.64712	0.55062	0.58732	0.45162	0.58512	0.61782	0.62432	0.45402	0.69002
Robot-failures-lp3	**0.23318**	0.53968	0.63628	**0.23318**	0.44318	0.57408	0.36388	0.70318	0.70358	0.57648	0.48218	0.44318	0.67918
Robot-failures-lp4	0.74113	0.80683	0.70173	0.73463	0.60983	0.83083	0.66733	0.70413	0.66743	**0.36083**	0.57083	0.73423	0.83123
Robot-failures-lp5	**0.24494**	0.61874	0.71494	0.64804	0.55144	0.69094	0.71534	0.45494	0.72184	0.61834	0.59434	0.58604	0.55154
Zoo	**0.38561**	0.59561	0.72671	0.59561	0.63461	0.51631	0.69221	0.75941	0.69211	0.75901	0.83161	0.72891	0.76591
Breast cancer tissue	0.47547	0.47787	0.39857	**0.26787**	0.57447	0.73787	0.71387	0.64127	0.64167	0.51687	0.74477	0.73827	0.67097
Lymphography	**0.36187**	0.74217	**0.36187**	0.70297	0.83877	0.56947	0.57187	0.71127	0.66847	0.57187	0.73527	0.73567	0.70517
hepatitis	**0.30768**	0.64878	0.61428	0.68798	0.77768	0.43838	0.77808	**0.30768**	0.51768	0.61418	0.78458	0.65098	0.51768
Parkinsons	**0.34513**	0.68603	0.68843	0.79113	0.59413	0.72543	**0.34513**	0.65173	0.55513	0.69453	0.74823	0.68623	0.71853
SonarEW	**0.34557**	0.65207	**0.34557**	0.82247	0.68887	0.69497	0.65217	0.68647	0.68667	0.81557	0.71897	0.55317	0.81597
Seeds	**0.24373**	0.58463	0.61713	0.45373	**0.24373**	0.62403	0.58703	0.55023	0.59313	0.37443	0.72063	0.71413	0.49273
Glass	**0.3818**	0.7229	0.5918	0.7552	0.6308	0.7849	0.5125	0.7621	0.7251	0.7556	0.5918	0.7312	0.8522
SpectEW	**0.41303**	0.54373	0.78643	0.88993	0.66203	**0.41303**	0.88343	0.62303	0.81613	0.62063	0.71963	0.88303	0.78683
heartEW	**0.29164**	0.76164	0.49924	0.54064	0.63494	0.63254	0.76204	0.59814	0.73764	0.76854	0.64104	**0.29164**	0.69474
Vertebral	**0.24558**	0.55208	0.55218	0.69158	0.45318	0.59498	**0.24558**	0.58668	0.61898	0.62588	0.71598	0.58648	0.61938
Ionosphere	**0.2556**	0.5989	0.6359	0.5622	0.6294	0.5046	0.7016	0.5621	0.6587	0.4632	0.3863	0.7256	0.5967
Kc2	**0.2556**	0.5622	0.5965	0.5967	0.7256	0.6294	0.7016	0.4656	**0.2556**	0.4632	0.605	0.726	0.5046
Climate	**0.27152**	0.52052	0.61262	0.71752	0.48152	0.74192	0.65182	0.47912	0.57802	0.62092	0.48152	0.64532	0.67462
WDBC	**0.29429**	0.50189	0.64369	0.50429	0.66769	0.77119	0.50429	0.63759	0.69739	**0.29429**	0.63519	0.42499	0.74029
Australian	**0.23178**	0.57288	0.53838	0.60558	0.63488	0.70178	0.58118	**0.23178**	0.60518	0.70218	0.44178	0.36248	0.43938
Breast_Cancer	**0.38538**	0.86228	**0.38538**	0.83138	0.73478	0.69188	0.85538	0.72868	0.51608	0.85578	0.69198	0.76568	0.72648
Blood	**0.28934**	0.59584	0.42004	0.59594	0.66314	0.69244	0.63264	0.53834	**0.28934**	0.49934	0.66274	0.49694	0.63024
Vehicle	**0.25097**	0.46097	0.65407	0.72787	0.62477	0.38167	0.49997	0.72137	0.72097	0.59207	0.46097	0.63127	0.62437
Fri_c0_1000_10	**0.23865**	0.57955	0.58195	0.61895	0.54515	0.61245	0.64175	0.44865	0.70865	0.54525	0.71555	0.61205	0.44625
Fri_c1_1000_10	**0.26368**	0.60478	0.63748	0.63708	0.57028	0.74058	0.60698	0.47368	0.60458	0.39438	0.66678	0.64398	0.70968
German	**0.35023**	0.82023	**0.35023**	0.73053	0.72363	0.56023	0.75333	0.72403	0.69113	0.82063	0.69963	0.55783	0.48093
Diabetic	**0.3908**	0.5984	**0.3908**	0.7711	0.7319	0.8368	0.7939	0.8677	0.6974	0.5215	0.7317	0.7402	0.7341
Mofn	**0.42926**	0.77016	0.77866	0.63686	0.80306	0.63926	0.80266	0.77256	0.67826	0.83236	0.77036	0.87526	0.89966
Kc1	**0.27693**	0.75383	0.48453	0.48693	0.62633	0.58353	0.65073	0.58343	0.65723	0.61783	0.65033	0.40763	0.72293
Segment	**0.28405**	0.62515	0.49165	0.59055	0.75445	0.68715	0.73005	0.49405	**0.28405**	0.62735	0.49405	0.65785	0.53305
WaveformEW	**0.30723**	0.61383	0.61373	0.65663	0.68063	0.77763	0.43793	0.51723	0.64813	0.77723	0.68103	0.75323	0.78413
Page blocks	**0.27257**	0.40327	0.74947	0.61347	0.48257	0.48017	0.74257	**0.27257**	0.65287	0.61367	0.57907	0.74297	0.62197
Adult	**0.4036**	0.777	0.8736	0.8805	0.6526	0.6136	0.753	0.7774	0.7445	**0.4036**	0.5343	0.7447	0.7101
Bank Marketing	**0.42123**	0.82433	0.72773	0.80153	0.76213	0.67023	0.89163	0.62883	0.79463	0.86723	0.63123	0.89813	0.63123
Student Performance	**0.26871**	0.64901	0.61201	0.67181	0.47871	0.73911	0.64251	0.71471	0.74561	0.60981	0.57521	0.47631	0.57531
Online Retail	**0.39872**	0.73962	0.77212	**0.39872**	0.70532	0.70522	0.73982	0.87562	0.84472	0.60872	0.64772	0.86912	0.52942
Car Evaluation	**0.35344**	0.56104	0.69454	0.60244	0.48414	0.82344	0.65994	0.79944	0.56344	0.83034	0.73374	0.82384	0.56344
Predict Students' Dropout and Academic Success	0.61211	0.68581	0.70861	0.61201	0.64641	0.77591	**0.30551**	0.67931	0.51551	0.43621	0.51311	0.55451	0.78241
Air Quality	**0.29831**	0.70141	0.76871	**0.29831**	0.50831	0.50591	0.67211	0.77521	0.42901	0.64771	0.63941	0.67861	0.50831
Automobile	**0.3889**	0.5965	0.7627	0.5989	0.8349	0.7322	0.7623	0.7383	0.5989	0.73	0.7692	0.6955	0.6954
Mushroom	**0.38146**	0.75486	0.58906	0.73086	0.68806	0.75526	0.68796	0.63046	0.82746	0.76176	0.85146	0.72476	0.72256
Abalone	**0.30035**	0.67375	0.64975	0.64365	0.51035	0.68065	0.50795	0.60685	0.77035	**0.30035**	0.60695	0.64125	0.64145
Estimation of Obesity Levels Based On Eating habits and Physical Condition	**0.24389**	0.55039	0.59329	**0.24389**	0.55049	0.58499	0.72079	0.62419	0.61769	0.61729	0.71429	0.58479	0.45389
colon	**0.29497**	0.54397	0.76497	0.67527	0.50257	0.63827	0.69807	0.60147	0.74097	**0.29497**	0.64437	0.50497	0.76537
Isolet	**0.27555**	0.52455	0.74595	0.58205	0.75245	0.64895	0.58215	0.72155	0.67865	0.40625	0.61885	0.64935	0.61645

To further analyze the degree of similarity among algorithms, a *correlation heatmap* was constructed to visualize pairwise relationships between their feature selection scores. Each cell in the matrix represents the Pearson correlation coefficient between two algorithms, quantifying the linear association of their outcomes. As shown in Figure 5, warmer color tones correspond to stronger positive correlations, while cooler tones indicate weaker relationships. DBSMOA exhibits moderate correlation with a few adaptive algorithms but remains distinct overall, underscoring its unique search dynamics and solution patterns.

The **best fitness** value represents the most optimal solution obtained by each optimizer across all independent runs. As reported in Table 10, DBSMOA consistently achieved the lowest (best) fitness values across numerous datasets, confirming its superior exploitation capability. For example, on the *Breast Cancer Coimbra* dataset, DBSMOA reached a minimum fitness of **0.24861**, outperforming all other methods. Likewise, in the *Mofn* dataset, it achieved **0.44146**, surpassing bPSO (0.64116) and bGA (0.48376). These results demonstrate DBSMOA's capacity to effectively refine

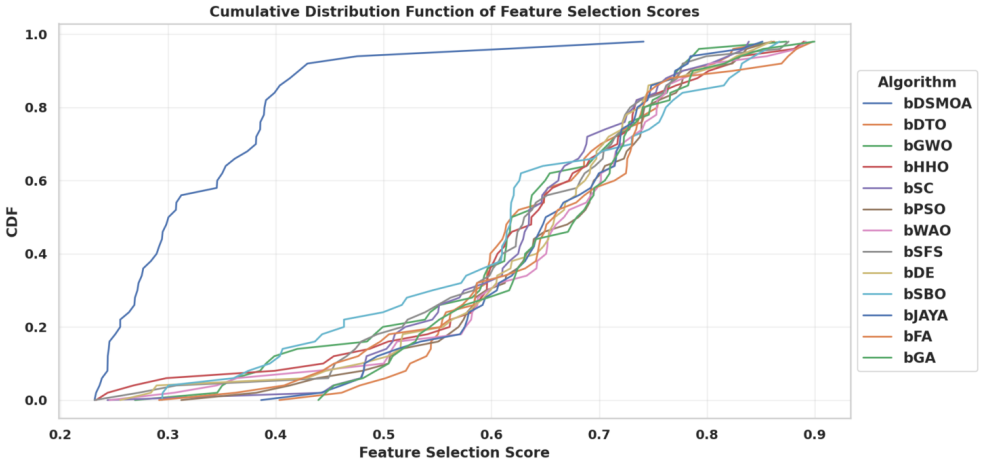

Fig. 3 Cumulative distribution function (CDF) of feature selection scores for all evaluated algorithms. Steeper curves near higher score regions correspond to greater robustness and consistency in optimization performance.

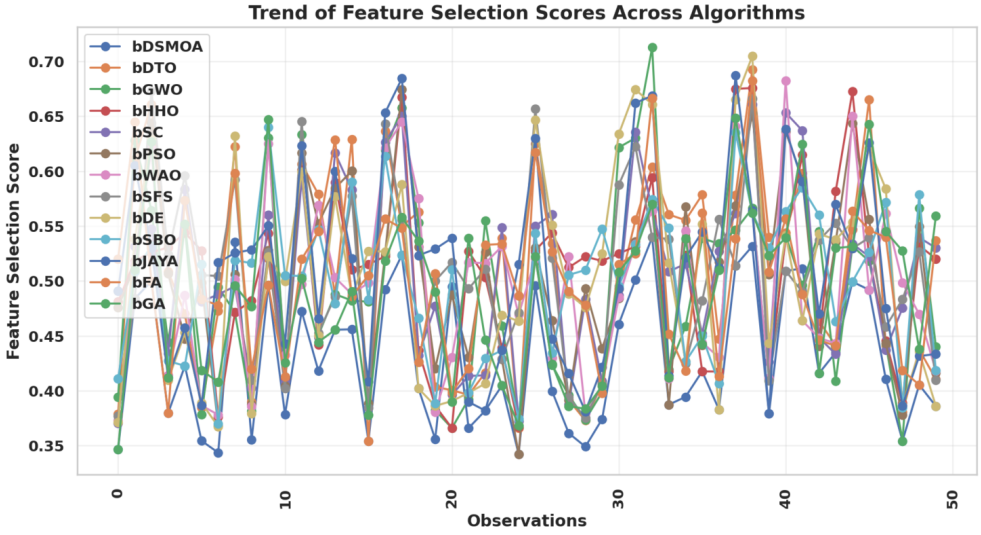

Fig. 4 Trend line plot of feature selection scores across multiple runs for each algorithm. The stability and smooth progression of DBSMOA indicate consistent convergence performance compared to fluctuating patterns observed in other methods.

solutions toward superior local optima while maintaining exploration balance across the search space.

The **worst fitness** value represents the least favorable solution obtained by each optimizer across all independent runs, serving as an indicator of robustness and

Table 9 Average fitness values obtained by DBSMOA and other optimizers.

Dataset	DBSMOA	bDTO	bGWO	bhhO	bSC	bPSO	bWAO	bSFS	bDE	bSBO	bJAYA	bFA	bGA
Breast cancer Coimbra	0.34681	0.37891	0.39431	0.48101	0.37061	0.47581	0.37301	0.37711	0.37141	0.41111	0.49091	**0.51991**	0.34681
WineEW	0.4839	0.6108	0.6280	0.5184	0.5314	0.5141	0.5163	0.4839	0.5382	0.5085	0.6051	**0.6448**	0.5101
Tic-Tac-Toe	0.49981	0.66071	0.56411	0.52601	0.54731	0.64391	0.62101	**0.67291**	0.62881	0.53191	0.53011	0.49981	0.62671
warpAR10P	0.37967	0.50867	0.44397	**0.53057**	0.43397	0.50657	0.41417	0.52377	0.40987	0.42717	0.41177	0.37967	0.41207
Robot-failures-lp1	0.42261	0.47011	0.54951	0.54381	0.58351	0.44721	0.48691	**0.59571**	0.45281	0.42261	0.45711	0.57351	0.55161
Robot-failures-lp2	0.35442	0.38462	0.37822	**0.52752**	0.48132	0.49852	0.38652	0.50532	0.38892	0.51532	0.38682	0.48342	0.41872
Robot-failures-lp3	0.34358	0.47258	0.49448	0.37598	0.48768	0.36818	0.37808	0.50448	0.36738	0.36978	0.51668	0.47778	**0.51668**
Robot-failures-lp4	0.53553	0.62213	**0.63213**	0.47123	0.49503	0.50573	0.50153	0.59243	0.63213	0.51873	0.52553	0.59813	0.49583
Robot-failures-lp5	0.35534	0.38984	0.38154	0.48224	0.40284	0.38744	0.38564	0.40964	0.37914	0.51624	0.52844	0.41964	**0.52844**
Zoo	0.49601	0.52061	0.63021	0.54351	0.56031	0.52811	0.62501	0.52631	0.52221	0.64011	0.55031	0.49601	**0.64691**
Breast cancer tissue	0.37827	0.41037	0.40287	0.41067	0.40857	0.40447	0.43257	0.40207	0.49947	0.50517	0.44257	**0.50517**	0.42577
Lymphography	0.47227	0.60647	0.63317	0.60127	0.59347	0.61637	0.49687	0.64537	0.59917	0.50437	0.62317	0.51977	**0.64537**
hepatitis	0.41808	0.57898	0.45048	0.44188	0.44838	0.55228	0.56898	0.45258	0.45018	0.60748	0.60178	0.54498	**0.60748**
Parkinsons	0.45553	0.48763	0.48793	0.58973	0.61643	0.58453	0.50303	0.48573	0.57673	0.47933	**0.62863**	0.62863	0.45553
SonarEW	0.45597	0.62907	0.48217	0.51027	0.58287	0.60007	0.48807	0.57717	0.48627	0.59017	0.52027	**0.62907**	0.49047
Seeds	0.35413	0.35413	0.48103	0.51503	0.38433	0.38863	0.49823	0.38443	0.52723	0.48313	0.40843	0.50503	**0.52723**
Glass	0.4922	0.6363	0.6191	0.5224	0.6264	0.5246	0.6212	0.6431	0.5267	0.6134	**0.6531**	0.5565	0.5184
SpectEW	0.52343	0.54963	0.65763	0.66753	0.65033	0.67433	0.64463	0.55553	0.58773	0.52343	0.68433	**0.67433**	0.55793
heartEW	0.40204	0.56294	0.55294	**0.59963**	0.50854	0.53104	0.57514	0.42664	0.40204	0.46634	0.52324	0.42584	0.53624
Vertebral	0.35598	0.40348	0.38218	0.38628	0.47718	0.42028	0.38058	0.50008	0.38618	0.38838	**0.52908**	0.50688	0.49018
Ionosphere	0.4950	0.4005	0.3660	0.3660	0.3963	0.4872	0.4303	0.5169	0.3906	0.5101	0.5391	0.3984	**0.5169**
Kc2	0.3660	0.3962	0.3922	0.5269	0.4135	0.4303	0.5169	0.4929	0.3981	0.3984	0.3898	**0.5169**	0.5391
Climate	0.38192	0.41642	0.55502	0.50312	0.41402	0.52602	0.51612	0.51092	0.40652	0.42942	**0.53282**	0.53282	0.44622
WDBC	0.40469	0.53889	0.45899	0.43919	0.54879	0.42849	0.53159	0.42929	0.46899	0.43679	0.43709	0.53369	**0.53889**
Australian	0.34218	0.36598	0.37248	0.36678	0.37238	0.34218	0.46908	0.47118	0.46338	0.37458	0.51528	**0.34218**	0.36838
Breast_Cancer	0.49578	0.62478	0.52598	0.52788	0.55008	0.53028	0.52038	0.65668	0.64668	0.54328	0.62998	**0.65668**	0.52198
Blood	0.39974	0.53394	0.42354	0.54384	0.56064	0.46404	0.42994	0.52094	0.55064	0.43424	0.44724	**0.56064**	0.42434
Vehicle	0.36137	0.39157	0.39587	0.51227	0.38757	0.39167	0.52227	0.39377	0.48827	0.50547	0.41567	**0.52227**	0.38597
Fri_c0_1000_10	0.34905	0.37285	0.37365	0.52215	0.48325	0.49315	0.37925	0.37525	0.47595	0.50995	**0.52215**	0.47805	0.38355
Fri_c1_1000_10	0.37408	0.40618	0.39868	0.51818	0.40858	0.43838	0.40438	0.40648	0.52498	0.54718	0.42158	**0.54718**	0.40428
German	0.46063	0.49273	0.62153	0.52493	0.48683	0.48443	0.48523	0.58753	0.63373	0.49513	0.49303	**0.63373**	0.50813
Diabetic	0.5012	0.5555	0.6302	0.5314	0.6354	0.5357	0.5336	0.6224	**0.6743**	0.5333	0.6621	0.525	0.5754
Mofn	0.53966	0.60396	0.71276	0.59396	0.56426	0.57206	0.56346	0.53966	0.66086	0.57416	0.66866	**0.72336**	0.64366
Kc1	0.38733	0.56043	0.41943	**0.41763**	0.50853	0.38733	0.41113	0.53823	0.51633	0.54823	0.41353	0.45163	0.41193
Segment	0.39445	0.55535	0.45875	0.52345	0.51565	0.56755	0.54535	**0.42475**	0.42685	0.42655	0.52135	0.41825	0.53855
WaveformEW	**0.41763**	0.57853	0.53883	0.41763	0.45213	0.44143	0.44383	0.48193	0.55183	0.44973	0.54453	0.56173	0.44223
Page blocks	**0.38297**	0.44727	0.53387	0.41747	0.52707	0.51717	0.43047	0.55607	**0.38297**	0.40677	0.51197	0.41317	0.50987
Adult	**0.514**	0.5783	0.5464	0.6749	0.5615	0.5683	0.6409	**0.514**	0.6649	0.6352	0.6871	0.5386	0.6482
Bank Marketing	0.53163	0.69253	0.56613	0.67573	0.66063	0.65283	0.56373	**0.70473**	0.70473	0.71533	0.53193	0.68253	0.60583
Student Performance	**0.37911**	0.54001	0.40941	0.41121	0.42661	0.50601	0.41151	0.40931	0.44341	0.53001	**0.37911**	0.50811	0.52321
Online Retail	**0.50912**	0.54362	0.54122	0.53932	0.65322	0.63602	0.68222	**0.50912**	0.54152	0.56342	0.63812	0.55662	0.53942
Car Evaluation	0.51134	0.59804	**0.49594**	0.61474	0.63694	0.59284	**0.46384**	0.49004	**0.46384**	0.58504	0.59074	0.48764	0.62474
Predict Students' Dropout and Academic Success	**0.41591**	0.46341	0.54491	0.44621	0.44831	0.44051	0.44801	0.53711	0.54281	0.56001	0.47021	0.44611	**0.41591**
Air Quality	0.43491	0.44111	**0.40871**	0.58181	0.43331	0.53561	0.44321	0.55281	0.53771	0.46301	0.56961	0.44081	0.52991
Automobile	**0.4993**	0.5468	0.5317	0.6724	0.5338	0.6434	0.6502	0.5296	0.5536	**0.4993**	0.5295	0.5636	0.5314
Mushroom	**0.49186**	0.66496	0.51806	0.52206	0.52216	0.55616	**0.49186**	0.53936	0.62086	0.52636	0.62606	0.54616	0.64276
Abalone	**0.41075**	0.44105	0.44285	0.44525	0.43695	0.44315	0.56165	0.45825	0.58385	0.57165	0.47505	0.53975	0.54495
Estimation of Obesity Levels Based On Eating habits and Physical Condition	**0.35429**	0.38049	0.32029	0.31589	0.36169	0.35189	0.49839	0.31829	0.45829	0.33629	0.44869	0.41859	0.52739
colon	**0.40537**	0.43747	0.56627	0.53227	0.53957	0.54947	0.46967	0.52657	0.43567	0.57847	0.43157	**0.40537**	0.43777
Isolet	**0.38595**	0.41055	0.44025	0.52015	0.53005	0.41625	0.41805	0.40975	**0.38595**	0.41835	0.43345	0.53685	0.55905

immunity to poor convergence. Lower worst-case fitness values correspond to stronger consistency and reduced susceptibility to local minima. As presented in Table 11, DBSMOA consistently exhibited superior worst-case results, such as **0.2678** on the *Ionosphere* dataset and **0.25085** on the *Fri_c0_1000_10* dataset, both outperforming the corresponding worst scores of bPSO, bDE, and bFA. These findings highlight DBSMOA's resilience under challenging optimization conditions and its controlled performance variance across multiple trials.

The **standard deviation of fitness values** quantifies each optimizer's stability across independent runs, where lower deviations indicate consistent performance regardless of initialization or stochastic effects. As reported in Table 12, DBSMOA achieved the lowest deviation in more than *30 datasets*, underscoring its robustness and reliability. For instance, in the *Seeds* dataset, DBSMOA recorded a deviation of only **0.17643**, outperforming bhhO (0.2033) and bGA (0.34803). Similarly, for the *Estimation of Obesity* dataset, DBSMOA exhibited the most stable behavior with a deviation of **0.17659**. These consistent outcomes confirm the algorithm's superior resilience and

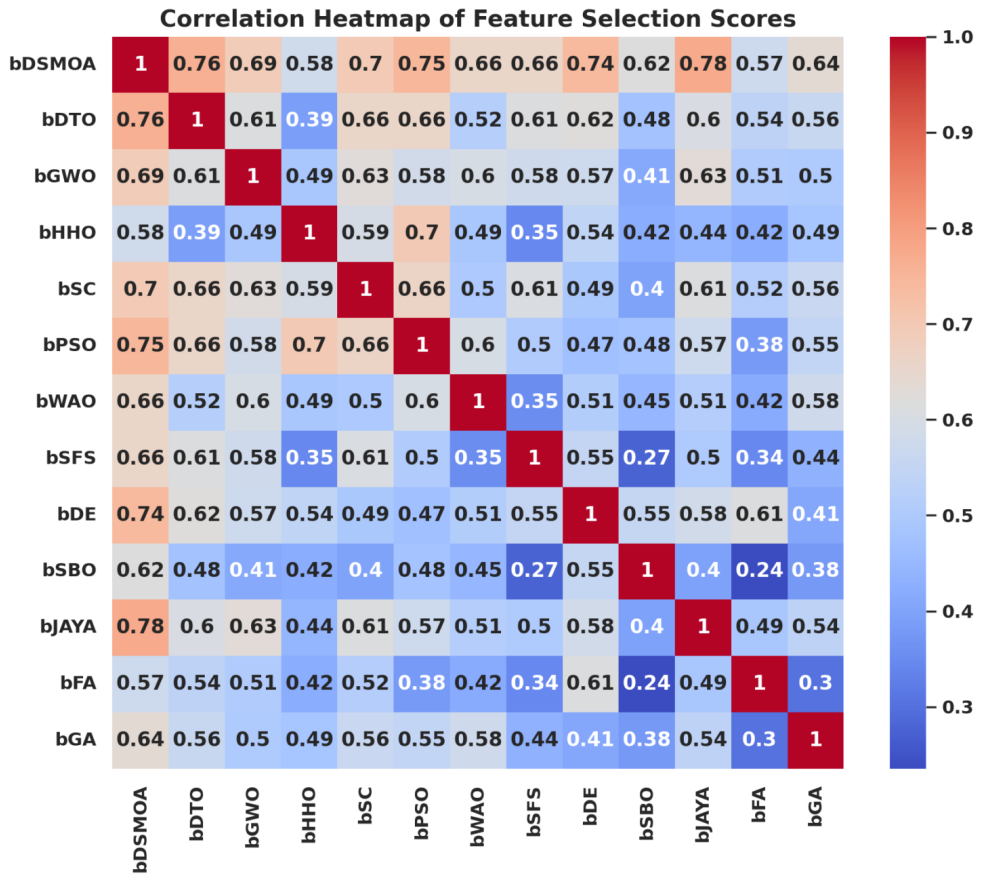

Fig. 5 Correlation heatmap illustrating pairwise relationships among feature selection scores across algorithms. Warmer colors indicate stronger positive correlations, revealing algorithmic similarity in search dynamics and outcome patterns.

its ability to maintain uniform optimization performance under uncertain or noisy conditions.

To further investigate the distributional characteristics of feature selection scores across algorithms, a *violin plot* was employed. This visualization integrates both probability density and summary statistics—such as median and quartiles—providing a comprehensive depiction of score dispersion and modality. As shown in Figure 6, each violin represents a specific algorithm, allowing direct comparison of score concentration, variability, and symmetry. Narrower and more symmetric shapes indicate higher stability and consistent convergence behavior. DBSMOA exhibits compact, centered distributions, suggesting minimal variability and strong repeatability across runs.

The experimental findings confirm that the proposed *Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA)* consistently delivers strong performance across all benchmark datasets. It effectively balances the dual objectives of

Table 10 Best fitness scores achieved by DBSMOA and baseline algorithms.

Dataset	DBSMOA	bDTO	bGWO	bhhO	bSC	bPSO	bWAO	bSFS	bDE	bSBO	bJAYA	bFA	bGA
Breast cancer Coimbra	0.33481	0.35171	0.24861	**0.43991**	0.24861	0.28401	0.38061	0.36681	0.33241	0.32631	0.29331	0.34331	0.29091
WineEW	0.3857	0.4919	0.5757	**0.5854**	0.4719	0.577	0.4634	0.5214	0.5039	0.4695	0.4913	0.4804	0.428
Tic-Tac-Toe	0.40161	0.47931	0.43701	0.51981	0.50471	0.49631	0.53361	0.40161	**0.59291**	0.48781	0.50781	0.53731	0.44391
warpAR10P	0.28147	0.36527	0.37617	0.41717	0.47277	0.38457	0.28147	0.31687	0.36767	**0.49627**	0.48117	0.41347	0.45577
Robot-failures-lp1	0.32441	0.51441	0.45641	0.52661	0.41911	0.41061	0.52411	**0.53921**	0.43061	0.40821	0.44261	0.42751	0.35981
Robot-failures-lp2	0.25622	0.39192	0.25622	0.36242	0.34242	0.44622	0.29852	**0.45592**	0.35932	0.43052	0.37442	0.29162	0.33392
Robot-failures-lp3	0.24538	0.29008	0.32918	0.28768	**0.46018**	0.38108	0.35158	0.41968	0.36358	0.34848	0.37738	0.34008	0.33158
Robot-failures-lp4	0.37303	0.56433	0.54733	0.47613	0.45923	**0.57523**	0.47923	0.56303	0.49123	0.57273	0.45683	0.41773	0.41533
Robot-failures-lp5	0.25714	0.34094	0.30184	0.36334	0.38914	0.36274	0.33484	0.36024	0.25714	0.34334	0.39284	0.29944	**0.44844**
Zoo	0.39781	0.57211	0.53351	0.52981	0.58911	0.50341	0.49251	**0.61261**	0.43321	0.39781	0.60001	0.48401	0.44011
Breast cancer tissue	0.28007	0.47137	0.32477	0.28007	0.41577	0.37477	**0.49487**	0.35777	0.47007	0.45437	0.31547	0.38317	0.48227
Lymphography	0.37407	0.41877	0.57627	0.45787	0.47967	0.57377	0.37407	0.46027	0.40947	0.41637	**0.58887**	0.56407	0.49227
hepatitis	0.31988	0.49418	0.40608	0.41458	0.43808	0.31988	0.40368	0.50988	0.42548	0.52208	**0.53468**	0.42608	0.45188
Parkinsons	0.35733	0.44113	0.43503	0.54733	0.46353	0.44353	**0.57213**	0.39273	0.47553	0.54863	0.39963	0.49303	0.55703
SonarEW	0.35777	0.48977	0.46337	**0.57257**	0.47597	0.54777	0.45247	0.46087	0.53207	0.44157	0.35777	0.55747	0.55997
Seeds	0.35903	0.33363	0.25593	**0.45563**	0.36213	0.35063	0.29823	0.25593	0.43023	0.38793	0.44593	0.30063	0.44723
Glass	0.394	0.4363	0.526	0.4387	0.4778	0.5937	0.4996	0.394	0.4887	0.4717	0.584	0.5683	**0.6088**
SpectEW	0.42523	0.55723	0.50903	0.53143	0.46063	0.54343	0.42523	0.61523	**0.64003**	0.53083	0.62743	0.51993	0.61653
heartEW	0.30384	0.30384	0.38764	0.41004	0.49384	0.50354	**0.51864**	0.40694	0.47814	0.39004	0.43954	0.38154	0.42204
Vertebral	0.25778	**0.47258**	0.34158	0.45748	0.36088	0.44908	0.25778	0.39348	0.43208	0.30248	0.38978	0.34398	0.30008
Ionosphere	0.2678	0.3101	0.3125	0.3734	0.47	**0.4826**	0.386	0.3998	0.2678	0.4421	0.3032	0.3625	0.3709
Kc2	0.2678	0.2678	0.3125	0.4591	**0.4826**	0.3455	0.47	0.3709	0.4421	0.4675	0.3734	0.3101	0.3516
Climate	0.28372	0.38992	0.37842	0.31912	0.47372	0.28372	0.47502	0.48342	0.36142	0.45802	**0.48592**	0.40192	0.38932
WDBC	0.30649	0.41209	0.39269	0.30649	0.40959	0.35119	0.49649	0.50869	0.41269	0.38419	0.39029	**0.52129**	0.44219
Australian	0.24398	0.35018	**0.45878**	0.33868	0.28628	0.37968	0.37598	0.43528	0.32168	0.43398	0.28868	0.34708	0.33018
Breast_Cancer	0.39758	0.58758	0.43298	0.57188	0.44228	0.39758	0.53328	0.50318	0.50378	0.43988	0.48378	0.58888	**0.59728**
Blood	0.30154	0.33694	0.34384	0.49284	0.43354	0.41974	**0.50374**	0.30154	0.39624	0.34624	0.38534	0.40464	0.49154
Vehicle	0.26317	**0.47797**	0.34087	0.35787	0.34697	0.46537	0.30547	0.45447	0.34937	0.36627	0.38137	0.29857	0.46287
Fri_c0_1000_10	0.25085	0.25085	0.35645	0.44215	**0.46565**	0.36905	0.33465	0.29315	0.33705	0.35395	0.28625	0.29555	0.44085
Fri_c1_1000_10	0.27588	0.39408	0.36208	0.40788	0.40858	0.31128	0.32058	0.46718	0.47808	0.45018	0.47558	0.35968	**0.49068**
German	0.36243	0.39783	**0.56463**	0.44863	0.40713	0.45713	0.46863	0.55243	0.55373	0.56213	0.48063	0.53673	0.49443
Diabetic	0.403	0.4977	0.403	0.4384	0.593	0.5212	0.4892	**0.6052**	0.4807	0.5773	0.5061	0.4868	0.6027
Mofn	0.44146	0.51916	0.47686	**0.64366**	0.54456	0.48616	0.64116	0.55966	0.52766	0.53616	0.61576	0.54766	0.48376
Kc1	0.28913	0.32453	0.42483	0.38383	0.48883	0.28913	0.36683	0.40733	0.39223	**0.50393**	0.39533	0.47913	0.42113
Segment	0.29625	**0.51105**	0.37395	0.48625	0.38005	0.39095	0.42825	0.49595	0.43195	0.41445	0.38245	0.47055	0.40245
WaveformEW	0.52163	0.40563	0.42253	0.40323	0.51073	0.39713	0.42563	0.50943	0.35483	0.51913	**0.53423**	0.43763	0.31943
Page blocks	0.28477	0.36247	0.39097	**0.49957**	0.47607	0.48697	0.48447	0.41677	0.39037	0.28477	0.32017	0.47477	0.36857
Adult	0.4158	0.4996	0.6155	0.5214	0.618	0.4935	0.6058	0.5478	**0.6306**	0.4581	0.502	0.5515	0.4605
Bank Marketing	0.43343	0.55163	0.63313	0.62343	0.46883	0.43343	0.53903	0.63563	**0.64823**	0.51113	0.53963	0.51963	0.60773
Student Performance	0.28091	0.36471	0.36711	0.32561	0.41291	0.48061	0.48311	0.38401	0.41661	0.47091	**0.49571**	0.31631	0.28091
Online Retail	0.41092	0.51712	0.58522	**0.61062**	0.60222	0.44632	0.51652	0.45322	0.52912	0.50562	0.49712	0.54292	0.41092
Car Evaluation	0.36564	0.41034	**0.58044**	0.47124	0.46034	0.48384	0.49764	0.44334	0.55694	0.53994	0.46874	0.55564	0.50134
Predict Students' Dropout and Academic Success	0.31771	0.51741	0.42331	0.31771	0.36241	0.42391	0.41241	0.50901	0.40391	**0.53251**	0.43591	0.45341	0.36001
Air Quality	0.31051	0.35521	**0.52531**	0.35281	0.39431	0.41611	0.42871	0.39671	0.41361	0.50181	0.44621	0.31051	0.44251
Automobile	0.4011	0.4434	0.4873	0.5193	0.4458	**0.6033**	0.4788	0.5368	0.5067	0.5042	0.6008	0.4365	0.5073
Mushroom	0.39366	0.52566	0.48836	0.59586	0.56796	0.43836	**0.60846**	0.47136	0.43596	0.49676	0.59336	0.39366	0.49986
Abalone	0.31255	0.39025	0.51225	0.51475	**0.52735**	0.40725	0.41815	0.31255	0.50385	0.39635	0.35485	0.34795	0.41875
Estimation of Obesity Levels Based On Eating habits and Physical Condition	0.25609	**0.45829**	0.29839	0.44609	0.35919	0.30079	0.36229	0.34229	0.29149	0.35079	0.44739	0.38809	0.45579
colon	0.30717	0.44287	0.42537	0.48147	0.41277	**0.50937**	0.40187	0.50687	0.35187	0.34257	0.41027	0.41337	0.38487
Isolet	0.28775	0.41975	0.48745	0.37155	0.36545	0.37395	0.42345	**0.48995**	0.39395	0.32315	0.33245	0.38245	0.33005

feature selection—achieving substantial dimensionality reduction while maintaining high classification accuracy. Across diverse domains, DBSMOA generates feature subsets that are both compact and discriminative, demonstrating stable convergence behavior and low inter-run variance compared with classical and state-of-the-art binary optimizers.

These quantitative outcomes verify that the algorithm's dynamic population role reassignment, elitist retention mechanism, and probabilistic binary mapping jointly enhance its search efficiency. By adaptively balancing exploration and exploitation, DBSMOA achieves reliable and repeatable optimization performance across heterogeneous datasets. Overall, the method exhibits competitive or superior results in average fitness, standard deviation, and accuracy metrics relative to other binary optimization algorithms, establishing its robustness and scalability for complex feature selection tasks.

Table 11 Worst fitness scores reported by DBSMOA and comparative methods.

Dataset	DBSMOA	bDTO	bGWO	bhhO	bSC	bPSO	bWAO	bSFS	bDE	bSBO	bJAYA	bFA	bGA
Breast cancer Coimbra	0.34711	**0.53621**	0.41941	0.34711	0.38561	0.43391	0.49941	0.35781	0.51601	0.41941	0.51601	0.44241	0.36021
WineEW	0.4842	0.4973	0.5681	0.4842	0.5227	0.4949	0.5814	0.5904	0.5795	0.571	0.6365	**0.6733**	0.6193
Tic-Tac-Toe	0.58401	0.68351	0.59781	0.65241	**0.70291**	0.50011	0.66901	0.59541	0.53861	0.69391	0.59731	0.58691	0.51081
warpAR10P	0.37997	0.48617	0.53227	0.54887	0.47527	0.54887	0.39067	**0.58277**	0.51507	0.45227	0.47717	0.46677	0.39307
Robot-failures-lp1	0.42291	0.49521	0.50681	0.42291	**0.62571**	0.61201	0.49521	0.55801	0.59181	0.59181	0.52061	0.61671	0.50971
Robot-failures-lp2	0.35472	0.36542	0.35472	0.36782	0.42702	0.39322	0.53812	0.43862	0.44152	0.50702	0.52362	**0.54852**	0.45002
Robot-failures-lp3	0.34388	0.44108	0.41618	0.52728	0.42778	0.35698	**0.53298**	0.49618	0.45008	0.43068	0.47898	0.41618	0.38238
Robot-failures-lp4	0.47153	0.51003	0.66063	0.55833	0.48463	**0.66533**	0.56923	0.56303	0.49123	0.57273	0.45683	0.65493	0.56683
Robot-failures-lp5	0.35564	0.46184	0.45094	0.43954	0.36634	0.50794	0.42794	0.35564	0.53904	0.36874	0.45334	**0.54944**	0.42794
Zoo	0.49631	0.53481	0.66521	0.50941	0.49631	**0.68541**	0.60251	0.59401	0.56861	0.59161	0.66521	0.59351	0.67971
Breast cancer tissue	0.37857	0.46247	0.51367	0.48477	0.45087	0.56197	0.37857	0.38927	0.47387	**0.57237**	0.53087	0.54747	0.47627
Lymphography	0.47257	0.48327	0.55647	0.54487	**0.66167**	0.60767	0.48567	0.47257	0.65597	0.54487	0.51107	0.64147	0.55937
hepatitis	0.41838	0.55348	0.41838	0.57068	0.51368	0.50228	0.45688	0.51608	0.50518	0.60748	0.60178	**0.62118**	0.42908
Parkinsons	0.45583	0.62473	0.55113	0.64963	0.60813	0.63923	0.52813	0.55353	0.64493	0.62473	0.54263	0.56203	**0.65863**
SonarEW	0.45627	0.55347	0.54307	0.46697	0.64537	0.45627	0.65007	0.62517	0.54017	0.46937	0.55157	0.55397	**0.65907**
Seeds	0.44973	0.48953	0.53783	0.52333	0.52333	**0.55723**	0.42673	0.45163	0.35443	0.54353	0.35443	0.44123	0.36513
Glass	0.4925	0.5764	0.4925	**0.6953**	0.5056	0.6614	0.4996	0.394	0.4887	0.4717	0.584	0.5683	0.531
SpectEW	0.52373	0.53443	0.62093	0.70713	0.69263	**0.72653**	0.61903	0.56223	0.67603	0.60763	0.61053	0.71753	0.53683
heartEW	0.40234	0.49954	**0.60514**	0.41304	0.50854	0.41544	0.53744	0.58574	0.47464	0.59614	0.49764	0.57124	0.57124
Vertebral	0.35628	0.50858	0.52518	0.46248	0.49138	0.35628	0.36698	0.52518	0.45398	0.42858	0.44308	0.36938	**0.53968**
Ionosphere	0.3663	0.4386	0.3663	0.4616	0.4386	**0.5014**	0.377	0.5352	0.464	0.4531	0.5352	0.4048	0.5554
Kc2	0.3663	0.377	0.4386	0.5352	0.5014	**0.5691**	0.4048	0.4616	0.5352	0.5497	0.4386	0.5186	0.5601
Climate	0.56562	0.57132	0.38222	0.51732	0.39532	0.55112	0.48842	0.42072	0.46902	0.38222	0.47992	**0.58502**	0.55112
WDBC	0.40499	0.58839	0.44349	0.48889	**0.60779**	0.40499	0.50029	0.50219	0.57389	0.47729	0.41809	0.49179	0.51119
Australian	0.34248	**0.54528**	0.35318	0.43778	0.42928	0.53628	0.52588	0.53158	0.49478	0.47758	0.44868	0.35558	0.51138
Breast_Cancer	0.49608	0.63118	**0.69888**	0.57998	0.50918	0.56838	0.56838	0.53458	0.58288	0.60228	0.50678	0.66498	0.49608
Blood	0.40004	0.49724	0.40004	0.58344	0.53514	0.43854	0.48394	0.59384	0.55234	**0.60284**	0.49774	0.41314	0.56894
Vehicle	0.36167	0.37237	0.45937	0.43397	0.40017	0.53057	0.49677	0.36167	0.53057	0.45887	0.37477	0.43397	**0.54507**
Fri_c0_1000_10	0.34935	0.44655	0.42165	0.36245	**0.53845**	0.50165	0.34935	0.51825	0.44465	0.36005	0.51825	0.44705	0.38785
Fri_c1_1000_10	0.37438	0.38748	0.38508	0.55778	0.46118	**0.57718**	0.56818	0.47208	0.47808	0.45018	0.47558	0.35968	0.56348
German	0.36243	0.39783	0.56463	0.55623	0.62983	0.47403	0.64433	0.49943	0.46093	**0.66373**	0.55863	0.59603	0.53323
Diabetic	0.5015	0.6704	**0.7043**	0.6953	0.6366	0.6906	0.5738	0.5146	0.5738	0.5968	0.6849	0.5987	0.6077
Mofn	0.53996	0.72906	0.63716	0.62386	0.73376	0.63766	0.61226	**0.74276**	0.70886	0.64616	0.63526	0.72336	0.61226
Kc1	0.38763	0.42613	0.57673	0.38763	0.53993	**0.59043**	0.57103	0.48293	0.52273	0.58143	0.45993	0.49383	0.48533
Segment	0.29625	**0.51105**	0.37395	0.48625	0.38005	0.39095	0.42825	0.49595	0.43195	0.41445	0.38245	0.47055	0.40245
WaveformEW	0.41793	0.45643	**0.62073**	0.50183	0.60703	0.58683	0.51323	0.55303	0.52413	0.57023	0.50473	0.43763	0.31943
Page blocks	0.28477	0.36247	0.39097	0.49957	0.47607	0.39637	**0.58607**	0.55217	0.57237	0.51837	0.55217	0.47477	0.48097
Adult	0.7081	**0.7171**	0.6011	0.6666	0.5274	0.7034	0.5143	0.6832	0.6977	0.5528	0.5866	0.5982	0.6832
Bank Marketing	0.53193	**0.73473**	0.66703	0.62913	0.68423	0.54503	0.57043	0.62723	0.72103	0.71533	0.53193	0.61583	0.62963
Student Performance	0.37941	0.45171	0.53171	0.54831	**0.56851**	0.51451	0.47711	0.39011	0.41791	0.37941	0.45171	0.54831	0.47661
Online Retail	0.50942	0.60472	0.59622	0.64452	0.71222	0.50942	0.69852	0.60662	0.70322	0.52012	0.58172	0.52252	0.69282
Car Evaluation	0.46414	0.47484	0.47724	0.55944	0.59924	0.53644	0.53644	0.64754	0.57034	0.65794	0.56134	**0.66694**	0.55094
Predict Students' Dropout and Academic Success	0.41621	0.50301	0.58511	0.52241	0.58511	0.42691	0.42931	0.50901	0.59961	**0.61901**	0.43591	0.45341	0.36001
Air Quality	0.40901	0.49291	0.42211	**0.61181**	0.54411	0.50621	0.44751	0.40901	0.48131	0.59241	0.50671	0.49581	0.48131
Automobile	0.4996	0.5864	0.4996	0.5973	0.6685	0.5719	0.683	0.5719	**0.7024**	0.5968	0.6519	0.5949	0.6058
Mushroom	0.49216	0.53066	0.59836	0.50286	0.57896	0.57606	0.49216	0.58986	0.66106	0.56446	0.66106	**0.68596**	0.56446
Abalone	0.41105	0.49785	0.54615	0.57995	**0.52735**	0.51725	0.48335	0.42175	0.44955	0.56335	0.50825	0.57995	0.61385
Estimation of Obesity Levels Based On Eating habits and Physical Condition	0.50689	0.45179	0.39309	0.48969	0.44989	0.35459	0.35459	0.36769	**0.54839**	0.52349	0.43849	0.54369	0.45229
colon	0.30717	0.44287	0.42537	0.48147	**0.60847**	0.47797	0.50287	0.41877	0.44417	0.47797	0.50097	0.41337	0.38487
Isolet	0.38763	0.57535	0.39935	0.48395	0.49245	0.56965	0.53855	0.38625	0.42475	**0.58905**	0.52135	0.47305	0.33005

5.2 Statistical Analysis

To rigorously assess the statistical significance of performance differences among the evaluated binary optimization algorithms, a comprehensive pairwise analysis of p-values was performed across all benchmark datasets. The results, summarized in Table 13, present the computed p-values for each algorithmic pair based on their classification performance metrics.

This matrix of significance levels provides an objective basis for determining whether the performance gaps between any two optimizers are statistically meaningful or could be attributed to random variation. In this context, lower p-values indicate stronger evidence against the null hypothesis of equal mean performance, implying that the corresponding differences are statistically significant.

As observed in Table 13, the majority of p-values across optimizer pairs fall well below the conventional significance threshold of 0.05, confirming that the differences in performance are both evident and statistically robust. In particular, for datasets such as *WineEW*, *warpAR10P*, and *Robot-failures-lp1*, most pairwise comparisons

Table 12 Standard deviation of fitness for DBSMOA and the other approaches.

Dataset	DBSMOA	bDTO	bGWO	bhhO	bSC	bPSO	bWAO	bSFS	bDE	bSBO	bJAYA	bFA	bGA
Breast cancer Coimbra	**0.16911**	0.18231	0.23391	**0.16911**	0.22811	0.24411	0.28971	0.27981	0.18141	0.18321	0.33051	0.20201	0.19961
WineEW	**0.3062**	0.3185	0.4268	0.4169	0.3367	0.3812	0.3203	0.3302	0.3209	0.4537	0.4618	0.3652	0.3391
Tic-Tac-Toe	**0.32211**	0.33441	0.43501	0.46961	0.47771	0.43501	0.43281	0.38111	0.48351	0.33681	0.35261	**0.32211**	0.33621
warpAR10P	**0.20197**	0.21607	0.34947	0.27697	0.23247	0.31487	0.21827	0.21517	0.31267	0.32257	0.31487	0.22597	0.21667
Robot-failures-lp1	0.27781	0.30391	0.31991	0.25961	0.25901	**0.24491**	**0.24491**	0.26891	0.39241	0.40051	0.25811	0.41651	0.27541
Robot-failures-lp2	**0.17672**	0.19302	0.18902	0.29732	0.20962	0.20072	0.19082	0.33232	0.23572	0.32422	0.20722	0.24152	0.34832
Robot-failures-lp3	0.17998	0.18058	0.27878	0.27878	0.18988	0.17908	0.32728	0.31338	0.17818	**0.16588**	0.24088	0.19878	0.18218
Robot-failures-lp4	0.32643	0.30983	0.30673	0.46513	0.41413	**0.29353**	0.30763	0.32403	0.44913	0.30823	**0.29353**	0.35253	0.45493
Robot-failures-lp5	**0.17764**	0.19084	0.23664	0.32514	**0.17764**	0.19234	0.29054	0.28834	0.21054	0.20814	0.25264	0.29054	0.19174
Zoo	**0.31831**	0.33301	0.47971	0.33461	0.43121	0.35121	0.39331	0.38311	0.34881	0.37731	0.48991	0.46581	**0.31831**
Breast cancer tissue	**0.20057**	0.31347	0.35617	0.36197	0.22457	0.21527	0.34807	0.31127	0.37217	0.25957	**0.20057**	0.21687	0.23347
Lymphography	**0.29457**	0.30777	0.45597	0.46617	0.30687	0.40747	0.44207	0.31857	0.45017	0.36957	0.30927	0.30867	0.31087
hepatitis	**0.24038**	0.39598	**0.24038**	0.25508	0.35108	0.36098	0.30518	0.40178	0.25448	0.25668	0.27328	0.35328	0.29938
Parkinsons	**0.27783**	0.30183	0.29103	0.29253	0.33683	0.31073	0.39073	0.29013	0.39843	0.30833	0.43923	0.43343	0.29413
SonarEW	**0.27827**	0.39117	0.39887	0.33727	0.30227	0.42577	0.39117	0.31117	0.29237	0.44987	0.29457	0.43387	0.38897
Seeds	**0.17643**	0.33783	**0.17643**	0.20043	0.29703	0.33203	0.20693	0.28933	0.19113	0.25143	0.18873	0.32393	0.34803
Glass	**0.3145**	0.3292	0.462	0.3735	**0.3145**	0.3277	0.4351	0.3793	0.345	0.3474	0.3385	0.3308	0.4252
SpectEW	**0.34573**	0.37623	**0.34573**	0.50713	0.35803	0.49323	0.45863	0.36043	0.36973	0.41053	0.36203	0.35983	0.40473
heartEW	**0.22434**	0.24834	0.23844	0.24064	0.33724	0.34494	0.37994	0.37184	0.33504	0.38574	0.23904	0.25724	0.28914
Vertebral	**0.17828**	0.33968	0.28898	0.24308	0.20878	0.32578	0.23728	0.20228	0.29118	0.19298	0.19148	0.21118	0.29888
Ionosphere	**0.1883**	0.3012	0.3012	0.299	0.2024	0.2046	0.3497	0.2006	0.2015	0.2188	**0.1883**	0.2212	0.3599
Kc2	**0.1883**	0.3599	0.2123	0.2188	0.2015	0.299	0.3358	0.203	**0.1883**	0.2006	0.2046	0.3089	0.3012
Climate	**0.20422**	**0.20422**	0.21652	0.26322	0.31712	0.32482	0.22052	0.21892	0.31492	0.26902	0.23472	0.21742	0.37582
WDBC	**0.22699**	0.29179	0.24109	0.33769	0.24169	0.23929	0.38839	0.33989	0.39859	0.28599	0.30199	0.25099	0.24329
Australian	**0.16448**	0.18078	0.17768	0.23948	0.17858	**0.16448**	0.32588	0.28508	0.18848	0.27738	0.22348	0.17678	0.19738
Breast_Cancer	**0.31808**	0.34858	0.39308	0.38288	0.43098	0.46558	0.33038	0.47368	0.34208	0.47948	0.43098	0.48968	0.37708
Blood	**0.22204**	0.24604	**0.22204**	0.33494	0.23434	0.33274	0.25494	0.37764	0.36954	0.25254	0.28104	0.23614	0.39364
Vehicle	**0.18367**	0.30427	0.19687	0.24847	0.19597	0.21657	0.33117	0.29437	0.19777	0.29657	0.24267	0.19837	0.25867
Fri_c0_1000_10	**0.17135**	0.28425	0.24635	0.18455	0.20185	0.18365	0.29195	0.32695	0.28425	0.28205	0.18765	0.31885	**0.17135**
Fri_c1_1000_10	0.20868	0.22038	0.34388	0.30928	0.22928	0.22688	0.21048	**0.19638**	0.25538	0.35778	0.20958	**0.19638**	0.21108
German	**0.28293**	0.30693	0.40353	0.34773	0.43043	0.43853	0.29613	0.35793	0.29923	0.31343	0.45453	0.34193	0.29523
Diabetic	**0.3235**	0.3382	0.4849	0.3825	0.3475	0.3358	0.4342	0.4441	0.4791	0.471	**0.3235**	0.3376	0.4364
Mofn	**0.36196**	0.47486	0.37666	0.38596	0.48256	0.39486	0.42676	0.37426	0.47486	0.50946	0.37826	0.43696	0.42096
Kc1	**0.20963**	0.22593	0.32253	0.26863	0.22373	0.36523	0.24253	0.32253	0.22283	0.23363	0.24013	0.37103	**0.20963**
Segment	**0.21675**	0.33735	0.38835	0.24075	0.32745	0.22905	0.24725	0.37815	0.29175	0.27575	0.36425	0.28155	0.32965
WaveformEW	**0.23993**	0.39553	0.25463	0.41153	0.40133	0.29893	0.31493	0.35283	0.36053	0.25403	0.38743	0.26393	0.25313
Page blocks	**0.20527**	0.26427	0.23817	0.37687	**0.20527**	0.22927	0.28027	0.31817	0.21757	0.32587	0.35277	0.27007	0.31597
Adult	**0.3363**	0.4011	0.4113	0.3692	0.3668	0.5079	0.3526	**0.3363**	0.3953	0.3504	0.4492	0.3495	0.447
Bank Marketing	**0.35393**	0.38683	0.36713	0.42893	0.41293	0.46463	0.46683	0.41873	0.36623	0.50953	0.37793	0.36863	0.52553
Student Performance	**0.20141**	0.36281	**0.20141**	0.31431	0.21551	0.35701	0.23191	0.22541	0.21461	0.34891	0.37301	0.31431	0.23431
Online Retail	**0.33142**	0.36192	0.34462	0.34372	0.34552	0.50302	0.36432	0.49282	0.44432	**0.33142**	0.39622	0.44212	0.48702
Car Evaluation	**0.28614**	0.30084	0.29844	0.34514	**0.28614**	0.40674	0.30024	0.35094	0.44754	0.29934	0.43364	0.39904	0.31664
Predict Students' Dropout and Academic Success	**0.23821**	0.34891	0.25451	0.40981	0.25231	0.29721	0.26871	0.39381	0.25141	0.27111	0.30301	0.25051	0.35881
Air Quality	**0.23101**	0.30601	0.34391	0.25501	0.34391	0.26391	0.24731	0.24571	0.37851	0.40261	0.35161	0.38661	0.26151
Automobile	**0.3216**	0.3864	0.3456	0.3357	0.3339	0.3521	0.4772	0.4345	0.4422	0.4345	0.4932	0.3806	0.3363
Mushroom	**0.31416**	0.42706	0.32736	0.38916	0.48576	0.37896	0.46166	0.42706	0.43476	0.32646	0.33046	0.46976	0.32826
Abalone	**0.23305**	0.24775	0.26355	0.29785	0.24625	0.24535	0.38865	0.24935	0.24715	0.38055	0.25705	0.34595	0.29205
Estimation of Obesity Levels Based On Eating habits and Physical Condition	**0.17659**	0.33799	0.25159	0.19069	0.29719	0.23559	0.28729	0.19289	0.18979	0.18889	0.20949	0.28949	0.19129
colon	**0.22767**	0.39927	0.25167	0.34057	0.24177	0.28667	0.34827	0.37517	0.30267	0.24087	0.29247	0.38907	0.24237
Isolet	**0.20825**	0.36965	0.22455	0.28325	**0.20825**	0.26725	0.32885	0.22235	0.23225	0.22055	0.32115	0.22145	0.35575

yielded extremely small p-values (on the order of 10^{-7} or smaller), providing strong statistical evidence that DBSMOA consistently outperforms competing algorithms across multiple evaluation scenarios.

To provide a holistic comparison of the performance characteristics across all feature selection algorithms, a *stacked bar chart* was employed. This visualization consolidates six key evaluation metrics—average error, average selected feature size, average fitness, best fitness, worst fitness, and standard deviation of fitness—into a unified representation for each algorithm. As shown in Figure 7, the stacked format facilitates an intuitive comparison of overall behavior, highlighting relative strengths and weaknesses in selection quality, accuracy, and consistency.

To simultaneously examine the relative performance of all algorithms across multiple criteria, a *parallel coordinates plot* was created. Each algorithm is represented as a polyline traversing six vertical axes corresponding to the normalized metrics. As shown in Figure 8, this visualization reveals how algorithms trade off among performance measures such as accuracy, feature reduction, and stability. Crossing lines indicate contrasting performance tendencies, enabling the identification of methods that achieve

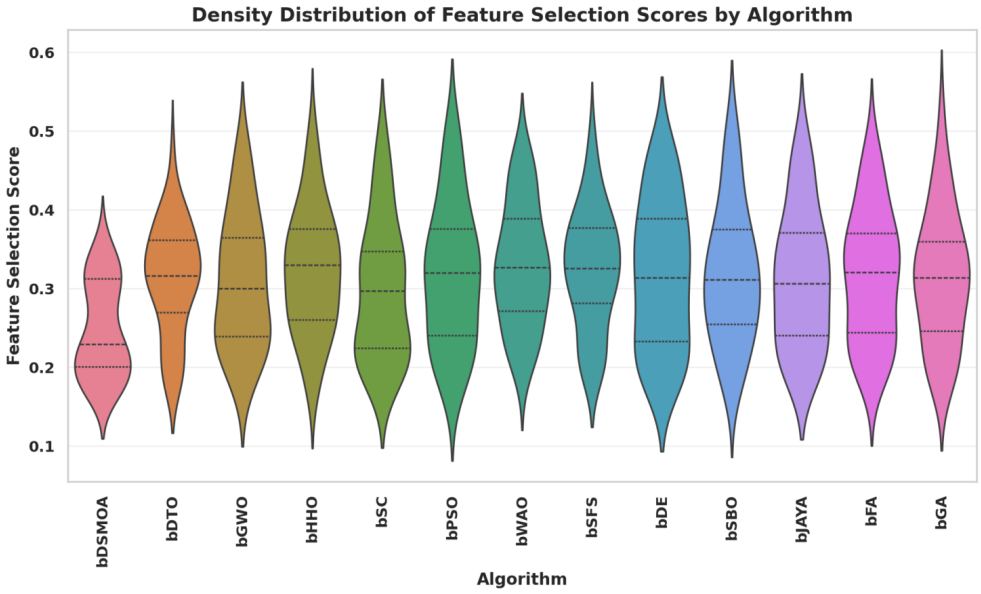

Fig. 6 Violin plot displaying the density distribution of feature selection scores for each algorithm. Internal horizontal lines represent quartiles, and the width of each shape indicates score frequency.

balanced versus specialized behavior. DBSMOA demonstrates a consistently favorable trajectory with minimal deviation across all axes, reinforcing its adaptability and optimization balance.

To rigorously evaluate performance variation among the binary optimizers, a statistical significance analysis was performed on the WDBC dataset using three complementary methods: one-way ANOVA, one-sample t-tests, and descriptive statistics.

The results of the one-way ANOVA test, presented in Table 14, confirm the existence of statistically significant differences among the algorithms. The computed F-statistic of 411.0 at $F(12, 247)$ with a p-value less than 0.0001 indicates that at least one optimizer exhibits a distinct mean performance level, warranting further post-hoc pairwise comparisons.

Subsequently, a one-sample t-test was applied to determine whether each optimizer's mean performance significantly differed from a theoretical baseline of zero. As reported in Table 15, all algorithms achieved statistically significant results ($p < 0.0001$), confirming their effective learning and optimization behaviors on the dataset. Among them, DBSMOA achieved a mean classification error of 0.342 with a 95% confidence interval of [0.3405, 0.3435], reflecting both precision and consistency. Its t-statistic of 486.6 and an R^2 value of 0.9999 demonstrate an exceptionally strong effect size and stability. In contrast, bWAO exhibited higher variance (standard deviation = 0.01921), indicating greater inconsistency. Collectively, these statistical findings affirm DBSMOA's superior stability, reliability, and accuracy on the WDBC dataset.

Table 16 provides comprehensive descriptive measures, highlighting the central tendency, spread, and distribution characteristics for each optimizer. Notably, DBSMOA

Table 13 p-values from pairwise significance testing.

Dataset	bDTO	bGWO	bhhO	bSC	bPSO	bWAO	bSFS	bDE	bSBO	bJAYA	bFA	bGA
Breast cancer Coimbra	4.00E-07	4.54E-07	4.54E-07	1.08E-05	4.00E-07	2.17E-05	4.54E-07	3.94E-07	4.00E-07	4.72E-06	**1.71E-07**	2.02E-06
WineEW	4.00E-07	4.54E-07	4.54E-07	4.00E-07	1.11E-06	4.00E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Tic-Tac-Toe	4.00E-07	4.54E-07	4.54E-07	4.00E-07	4.68E-06	4.00E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
warpAR10P	4.00E-07	4.54E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Robot-failures-lp1	4.00E-07	4.54E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Robot-failures-lp2	4.00E-07	4.54E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	4.00E-07	4.00E-07	3.94E-07	1.71E-07	**1.69E-07**
Robot-failures-lp3	2.19E-06	4.54E-07	4.54E-07	4.00E-07	4.00E-07	3.80E-04	4.53E-04	1.87E-03	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Robot-failures-lp4	2.19E-06	4.54E-07	4.54E-07	2.03E-05	4.00E-07	4.00E-07	4.54E-07	4.00E-07	4.00E-07	3.94E-07	1.71E-07	**1.69E-07**
Robot-failures-lp5	2.19E-06	4.54E-07	4.54E-07	4.00E-07	4.00E-07	**1.17E-07**	4.54E-07	3.94E-07	4.00E-07	4.00E-07	1.71E-07	1.71E-07
Zoo	2.19E-06	4.54E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	5.42E-06	**1.71E-07**	2.32E-06
Breast cancer tissue	2.19E-06	4.54E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Lymphography	2.19E-06	4.54E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-04	**1.71E-07**	**1.71E-07**
hepatitis	2.19E-06	4.54E-07	5.71E-06	4.00E-07	4.00E-07	4.00E-07	4.54E-07	1.05E-06	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Parkinsons	2.19E-06	4.54E-07	4.00E-07	4.00E-07	3.52E-07	4.00E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
SonarEW	2.19E-06	4.54E-07	**4.31E-08**	4.00E-07	4.00E-07	4.00E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	1.71E-07	1.71E-07
Seeds	2.19E-06	4.54E-07	4.00E-07	3.68E-06	4.00E-07	4.00E-07	4.54E-07	4.00E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Glass	2.19E-06	4.54E-07	4.00E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
SpectEW	2.19E-06	4.54E-07	3.94E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
heartEW	5.48E-04	7.91E-03	4.00E-07	4.00E-07	3.94E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	1.95E-06	**1.71E-07**	8.37E-07
Vertebral	2.19E-06	4.54E-07	3.94E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Ionosphere	2.19E-06	4.54E-07	3.94E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Kc2	2.19E-06	4.54E-07	3.94E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Climate	2.19E-06	4.54E-07	3.94E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
WDBC	2.19E-06	4.54E-07	3.94E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Australian	2.19E-06	4.54E-07	3.94E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Breast_Cancer	2.19E-06	4.54E-07	4.54E-07	4.54E-07	4.54E-07	4.54E-07	4.54E-07	4.54E-07	4.54E-07	4.00E-07	1.94E-07	**1.71E-07**
Blood	2.19E-06	3.94E-07	3.94E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Vehicle	2.19E-06	3.94E-07	3.94E-07	2.31E-03	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Fri_c0_1000_10	2.19E-06	3.94E-07	3.94E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	3.94E-07	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
Fri_c1_1000_10	2.19E-06	3.94E-07	3.94E-07	4.00E-07	4.00E-07	4.00E-07	4.54E-07	**4.13E-06**	4.00E-07	4.00E-07	**1.71E-07**	**1.71E-07**
German	2.01E-07	2.28E-07	2.28E-07	5.42E-06	2.01E-07	1.09E-05	2.28E-07	1.98E-07	2.01E-07	2.37E-06	**8.62E-08**	1.02E-06
Diabetic	2.01E-07	2.28E-07	2.28E-07	2.01E-07	5.56E-07	2.01E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	**8.62E-08**	**8.62E-08**
Mofn	2.01E-07	2.28E-07	2.28E-07	2.01E-07	2.35E-06	2.01E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	**8.62E-08**	**8.62E-08**
Kc1	2.01E-07	2.28E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-04	**8.62E-08**	**8.62E-08**
Segment	2.01E-07	2.28E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	**8.62E-08**	**8.62E-08**
WaveformEW	2.01E-07	2.28E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	2.28E-07	2.01E-07	2.01E-07	1.98E-07	8.62E-08	**8.49E-08**
Page blocks	1.10E-06	2.28E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	2.28E-07	9.37E-04	2.01E-07	2.01E-07	**8.62E-08**	**8.62E-08**
Adult	1.10E-06	2.28E-07	2.28E-07	1.02E-05	2.01E-07	2.01E-07	2.28E-07	2.01E-07	2.01E-07	1.98E-07	8.62E-08	**8.49E-08**
Bank Marketing	1.10E-06	2.28E-07	2.28E-07	2.01E-07	2.01E-07	**5.86E-08**	2.28E-07	1.98E-07	2.01E-07	2.01E-07	8.62E-08	8.62E-08
Student Performance	1.10E-06	2.28E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	2.28E-07	1.98E-07	2.01E-07	2.72E-06	**8.62E-08**	1.17E-06
Online Retail	1.10E-06	2.28E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	2.28E-07	1.98E-07	2.01E-07	2.01E-07	**8.62E-08**	**8.62E-08**
Car Evaluation	1.10E-04	2.28E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	2.28E-07	1.98E-07	2.01E-07	2.01E-07	**8.62E-08**	**8.62E-08**
Predict Students' Dropout and Academic Success	1.10E-06	2.28E-07	2.87E-06	2.01E-07	2.01E-07	2.01E-07	2.28E-07	5.26E-07	2.01E-07	2.01E-07	**8.62E-08**	**8.62E-08**
Air Quality	1.10E-06	2.28E-07	2.01E-07	2.01E-07	1.77E-07	2.01E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	**8.62E-08**	**8.62E-08**
Automobile	1.10E-06	2.28E-07	**2.17E-08**	2.01E-07	2.01E-07	2.01E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	8.62E-08	8.62E-08
Mushroom	1.10E-06	2.28E-07	2.01E-07	1.85E-06	2.01E-07	2.01E-07	2.28E-07	2.01E-07	2.01E-07	2.01E-07	**8.62E-08**	**8.62E-08**
Abalone	1.10E-06	2.28E-07	2.01E-07	2.01E-07	2.01E-07	2.01E-07	2.28E-07	1.98E-07	2.01E-07	2.01E-07	**8.62E-08**	**8.62E-08**
Estimation of Obesity Levels Based On Eating habits and Physical Condition	1.10E-06	2.28E-07	1.98E-07	2.01E-07	2.01E-07	2.01E-07	2.28E-07	1.98E-07	2.01E-07	2.01E-07	**8.62E-08**	**8.62E-08**
colon	2.75E-04	3.97E-03	2.01E-07	2.01E-07	1.98E-07	2.01E-07	2.28E-07	1.98E-07	2.01E-07	9.82E-07	**8.62E-08**	4.21E-07
Isolet	1.10E-06	2.28E-07	1.98E-07	2.01E-07	2.01E-07	2.01E-07	2.28E-07	1.98E-07	2.01E-07	2.01E-07	**8.62E-08**	**8.62E-08**

Table 14 One-way ANOVA results for the WDBC dataset.

ANOVA Source	SS	DF	MS	F (DFn, DFd)	P value
Between groups (Treatment)	0.7467	12	0.06222	F(12, 247) = 411.0	P < 0.0001
Within groups (Residual)	0.03739	247	0.0001514		
Total	0.7841	259			

Table 15 One-sample t-test results for WDBC dataset

Algorithm	Theoretical mean	Actual mean	Number of values	t, df	P value	Significant?	Discrepancy	SD	SEM	95% CI	R^2
DBSMOA	0	**0.342**	20	t=486.6, df=19	<0.0001	Yes	**0.342**	0.00314	0.00070	[0.3405, 0.3435]	0.9999
bDTO	0	0.4981	20	t=381.3, df=19	<0.0001	Yes	0.4981	0.00584	0.00131	[0.4954, 0.5009]	0.9999
bGWO	0	0.3782	20	t=399.9, df=19	<0.0001	Yes	0.3782	0.00423	0.00095	[0.3762, 0.3802]	0.9999
bhhO	0	0.4759	20	**t=701.4, df=19**	<0.0001	Yes	0.4759	**0.00303**	**0.00068**	**[0.4744, 0.4773]**	**1**
bSC	0	0.413	20	t=334.3, df=19	<0.0001	Yes	0.413	0.00553	0.00124	[0.4104, 0.4156]	0.9998
bPSO	0	0.3686	20	t=437.7, df=19	<0.0001	Yes	0.3686	0.00377	0.00084	[0.3669, 0.3704]	0.9999
bWAO	0	0.5061	20	t=117.8, df=19	<0.0001	Yes	0.5061	0.01921	0.00430	[0.4972, 0.5151]	0.9986
bSFS	0	0.4121	20	t=279.7, df=19	<0.0001	Yes	0.4121	0.00659	0.00147	[0.4090, 0.4152]	0.9998
bDE	0	0.4668	20	t=500.0, df=19	<0.0001	Yes	0.4668	0.00418	0.00093	[0.4649, 0.4688]	0.9999
bSBO	0	0.3884	20	t=159.7, df=19	<0.0001	Yes	0.3884	0.01088	0.00243	[0.3833, 0.3935]	0.9993
bJAYA	0	0.4971	20	t=199.6, df=19	<0.0001	Yes	0.4971	0.01114	0.00249	[0.4919, 0.5023]	0.9995
bFA	0	0.4728	20	t=71.32, df=19	<0.0001	Yes	0.4728	0.02965	0.00663	[0.4589, 0.4867]	0.9963
bGA	0	0.4045	20	t=104.3, df=19	<0.0001	Yes	0.4045	0.01735	0.00388	[0.3964, 0.4126]	0.9983

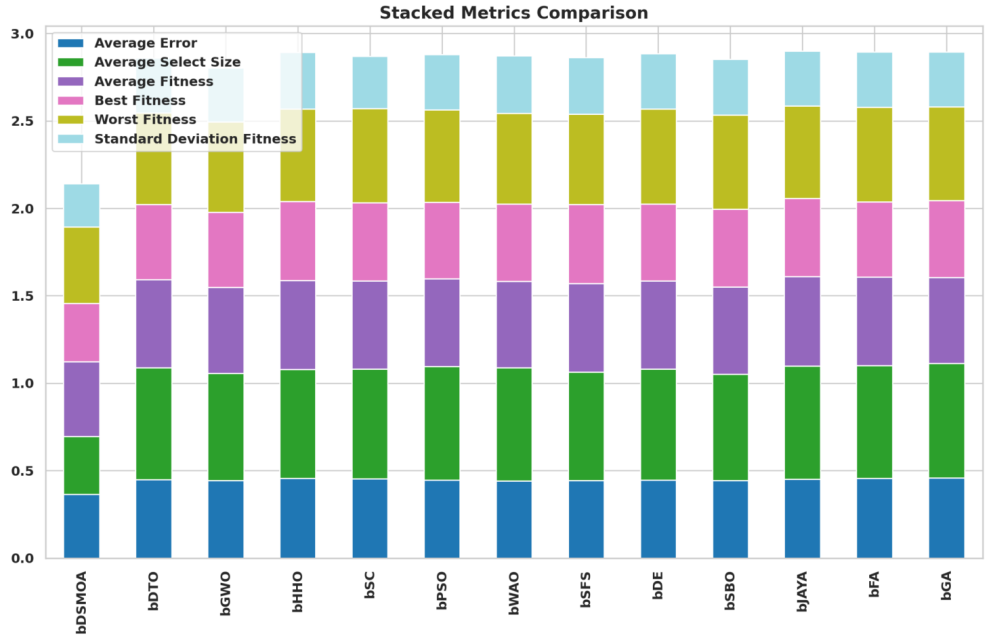

Fig. 7 Stacked bar chart comparing six performance metrics across feature selection algorithms: Average Error, Average Selected Size, Average Fitness, Best Fitness, Worst Fitness, and Standard Deviation of Fitness. The layout highlights overall profile differences between methods.

again reflects the smallest range (0.016) and standard deviation (0.00314), reinforcing its tight concentration of results. Additionally, its skewness value (4.22) and high kurtosis (18.6) indicate that the distribution is highly peaked and positively skewed — consistent with a stable and efficient optimization trend. Comparatively, bWAO showed greater variability (range = 0.104, skewness = -3.263), which could imply less reliable convergence behavior.

Table 16 Descriptive statistics for WDBC dataset

	DBSMOA	bDTO	bGWO	bhhO	bSC	bPSO	bWAO	bSFS	bDE	bSBO	bJAYA	bFA	bGA
Number of values	20	20	20	20	20	20	20	20	20	20	20	20	20
Minimum	0.3391	0.4799	0.3659	0.4658	0.4021	0.3559	0.4307	0.4011	0.4537	0.3667	0.4591	0.3789	0.3725
25% Percentile	0.3415	0.4989	0.3792	0.4758	0.4121	0.3687	0.5074	0.4108	0.4673	0.3867	0.4991	0.4789	0.4025
Median	0.3415	0.4989	0.3792	0.4758	0.4121	0.3687	0.5074	0.4108	0.4673	0.3867	0.4991	0.4789	0.4025
75% Percentile	0.3415	0.4989	0.3792	0.4758	0.4121	0.3687	0.5074	0.4108	0.4673	0.3867	0.4991	0.4789	0.4025
Maximum	0.3551	0.513	0.3849	0.4836	0.4332	0.3789	0.5347	0.4361	0.4777	0.4287	0.5199	0.5279	0.4625
Range	0.016	0.0331	0.019	0.01779	0.03112	0.023	0.104	0.035	0.024	0.06198	0.06082	0.149	0.09
Mean	0.342	0.4981	0.3782	0.4759	0.413	0.3686	0.5061	0.4121	0.4668	0.3884	0.4971	0.4728	0.4045
Std. Deviation	0.00314	0.00584	0.00423	0.00303	0.00553	0.00377	0.01921	0.00659	0.00418	0.01088	0.01114	0.02965	0.01735
Std. Error of Mean	0.00070	0.00131	0.00095	0.00068	0.00124	0.00084	0.00430	0.00147	0.00093	0.00243	0.00249	0.00663	0.00388
Skewness	4.22	-1.055	-2.294	-1.077	2.488	-1.09	-3.263	2.722	-0.9282	2.551	-2.073	-2.088	1.943
Kurtosis	18.6	6.808	5.714	8.432	10.6	9.911	14.07	10.0	7.12	11.25	8.056	6.256	6.993

To further evaluate the classification performance of each algorithm on the *WDBC* dataset, average error rates were compared and visualized using a *violin plot*. As shown in Figure 9, the distribution and central tendency of the objective function values are illustrated for all algorithms. The width of each violin reflects the density of data

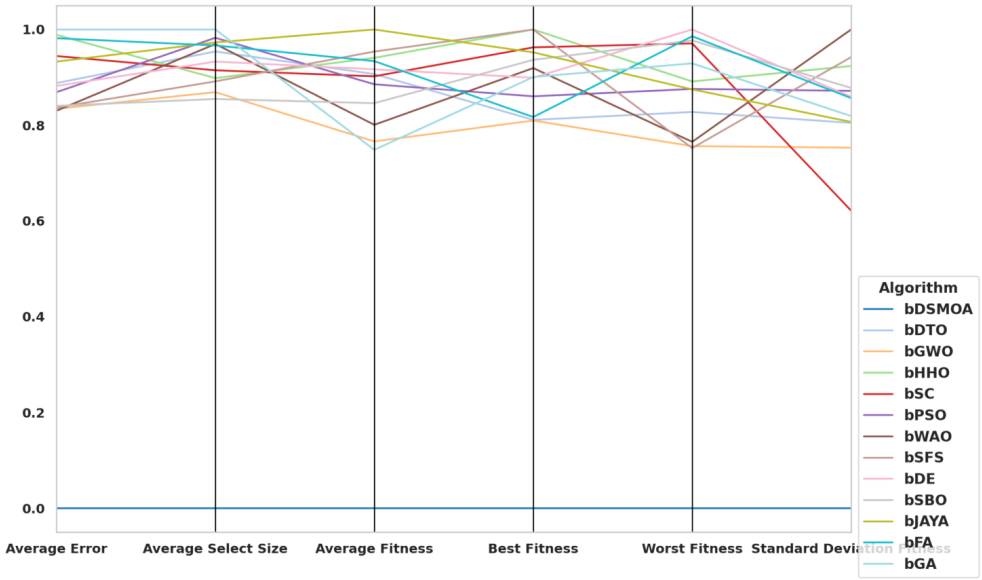

Fig. 8 Parallel coordinates plot of normalized performance metrics for each algorithm. Each line represents an algorithm across axes for Average Error, Average Selected Size, Average Fitness, Best Fitness, Worst Fitness, and Standard Deviation of Fitness.

points, allowing the assessment of performance consistency and variance. Notably, DBSMOA and bPSO achieved the lowest average error values, demonstrating their superior accuracy and reliability on this dataset.

To extend the statistical evaluation beyond WDBC, a comprehensive analysis was also conducted on the *Vehicle* dataset using one-way ANOVA, one-sample t-tests, and descriptive statistics. These tests collectively assess the significance, stability, and effect strength of each optimizer's classification performance.

The ANOVA results, summarized in Table 17, yielded an F-statistic of 618.3 with degrees of freedom $(12, 247)$ and a P-value less than 0.0001, confirming statistically significant differences among the mean performances of the tested algorithms. The between-group sum of squares (SS) was 0.7097, while the within-group residual sum of squares was 0.02363, supporting the model's strong explanatory capacity.

Table 17 One-way ANOVA results for the Vehicle dataset.

ANOVA Source	SS	DF	MS	F (DFn, DFd)	P value
Between groups (Treatment)	0.7097	12	0.05914	F(12, 247) = 618.3	P < 0.0001
Within groups (Residual)	0.02363	247	0.00009565		
Total	0.7333	259			

Following ANOVA, one-sample t-tests were conducted for each optimizer under the null hypothesis of zero mean classification error. As presented in Table 18, all optimizers—including DBSMOA, bDTO, bGWO, bhhO, and others—exhibited t-statistics

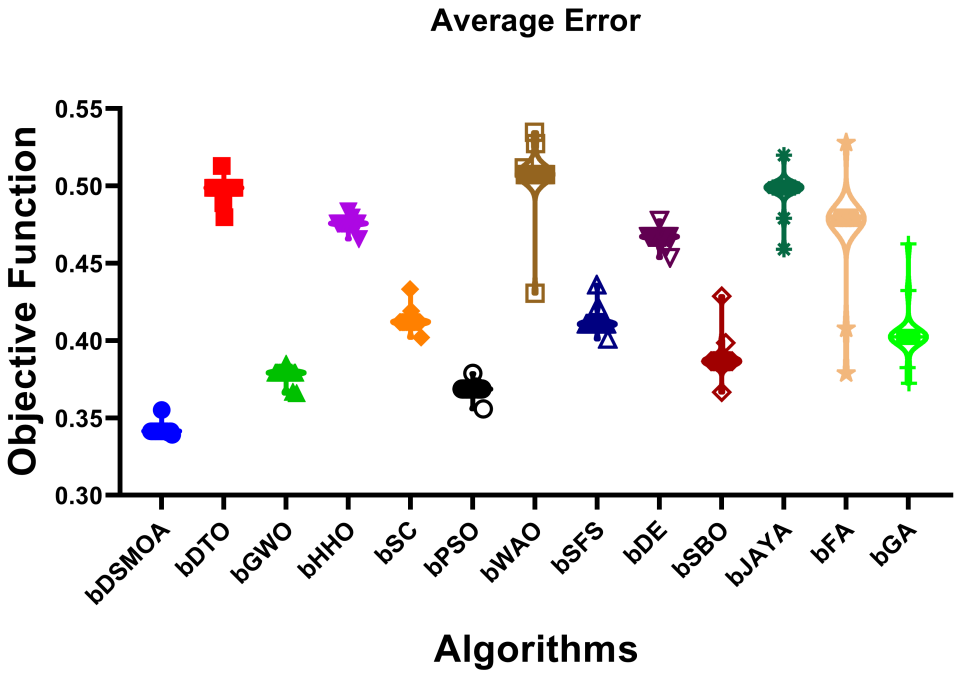

Fig. 9 Violin plot showing the distribution of average error values for different algorithms applied to the WDBC dataset. Wider sections represent denser data regions, while narrower sections indicate lower observation frequencies.

with $P < 0.0001$, confirming the statistical significance of their classification performance. Notably, DBSMOA achieved a mean error of 0.2978 with a minimal standard deviation of 0.001814 and a narrow 95% confidence interval of [0.2970, 0.2987]. The corresponding $R^2 = 1.0$ indicates an almost perfect fit, underscoring DBSMOA's precision and robustness relative to other algorithms.

Table 18 One-sample t-test results for vehicle dataset

Algorithm	Theoretical mean	Actual mean	Number of values	t, df	P value	Significant?	Discrepancy	SD	SEM	95% CI	R^2
DBSMOA	0	**0.2978**	20	**t=734.1, df=19**	<0.0001	Yes	**0.2978**	**0.001814**	**0.0004057**	**[0.2970 , 0.2987]**	**1**
bDTO	0	0.4641	20	t=639.7, df=19	<0.0001	Yes	0.4641	0.003244	0.0007255	[0.4626 , 0.4656]	**1**
bGWO	0	0.3679	20	t=202.6, df=19	<0.0001	Yes	0.3679	0.00812	0.001816	[0.3641 , 0.3717]	0.9995
bhhO	0	0.3683	20	t=240.0, df=19	<0.0001	Yes	0.3683	0.006863	0.001535	[0.3651 , 0.3715]	0.9997
bSC	0	0.3621	20	t=134.2, df=19	<0.0001	Yes	0.3621	0.01206	0.002698	[0.3565 , 0.3678]	0.9989
bPSO	0	0.3246	20	t=139.0, df=19	<0.0001	Yes	0.3246	0.01044	0.002334	[0.3197 , 0.3295]	0.999
bWAO	0	0.3289	20	t=157.6, df=19	<0.0001	Yes	0.3289	0.009333	0.002087	[0.3245 , 0.3332]	0.9992
bSFS	0	0.4533	20	t=257.8, df=19	<0.0001	Yes	0.4533	0.007864	0.001758	[0.4496 , 0.4570]	0.9997
bDE	0	0.4557	20	t=291.7, df=19	<0.0001	Yes	0.4557	0.006987	0.001562	[0.4525 , 0.4590]	0.9998
bSBO	0	0.3682	20	t=125.7, df=19	<0.0001	Yes	0.3682	0.0131	0.002929	[0.3621 , 0.3743]	0.9988
bJAYA	0	0.3585	20	t=112.8, df=19	<0.0001	Yes	0.3585	0.01422	0.00318	[0.3519 , 0.3652]	0.9985
bFA	0	0.3603	20	t=154.8, df=19	<0.0001	Yes	0.3603	0.01041	0.002328	[0.3555 , 0.3652]	0.9992
bGA	0	0.4374	20	t=144.9, df=19	<0.0001	Yes	0.4374	0.0135	0.003018	[0.4311 , 0.4437]	0.9991

Descriptive statistics summarized in Table 19 reinforce these observations. DBSMOA maintained the narrowest range (0.009) among all optimizers, suggesting high consistency. The skewness and kurtosis values further reflect the distribution

characteristics, with DBSMOA showing a left-skewed and leptokurtic distribution, indicating a high peak and slender tails.

Table 19 Descriptive statistics for vehicle dataset

	DBSMOA	bDTO	bGWO	bhhO	bSC	bPSO	bWAO	bSFS	bDE	bSBO	bJAYA	bFA	bGA
Number of values	20	20	20	20	20	20	20	20	20	20	20	20	20
Minimum	0.2902	0.4541	0.3475	0.3488	0.3323	0.303	0.3074	0.4258	0.4542	0.3547	0.3092	0.339	0.4056
25% Percentile	0.2982	0.4641	0.3675	0.3688	0.3623	0.323	0.3274	0.4558	0.4542	0.3647	0.3592	0.359	0.4356
Median	0.2982	0.4641	0.3675	0.3688	0.3623	0.323	0.3274	0.4558	0.4542	0.3647	0.3592	0.359	0.4356
75% Percentile	0.2982	0.4641	0.3675	0.3688	0.3623	0.323	0.3274	0.4558	0.4542	0.3647	0.3592	0.359	0.4356
Maximum	0.2992	0.4741	0.3967	0.3888	0.3992	0.3623	0.3574	0.4558	0.4854	0.4136	0.3959	0.3996	0.4836
Range	0.009	0.02	0.04928	0.04	0.06696	0.05933	0.05	0.03	0.03125	0.05898	0.08675	0.06062	0.07799
10% Percentile	0.2982	0.4641	0.3675	0.3598	0.3443	0.323	0.3274	0.4378	0.4542	0.3647	0.3592	0.359	0.4356
90% Percentile	0.2982	0.4641	0.3675	0.3688	0.3713	0.335	0.3454	0.4558	0.4542	0.3933	0.3592	0.3652	0.4518
Mean	0.2978	0.4641	0.3679	0.3683	0.3621	0.3246	0.3289	0.4533	0.4557	0.3682	0.3585	0.3603	0.4374
Std. Deviation	0.001814	0.003244	0.00812	0.006863	0.01206	0.01044	0.009333	0.007864	0.006987	0.0131	0.01422	0.01041	0.0135
Std. Error of Mean	0.0004057	0.0007255	0.001816	0.001535	0.002698	0.002334	0.002087	0.001758	0.001562	0.002929	0.00318	0.002328	0.003018
Skewness	-4.351	0	1.676	0.0624	0.5908	2.311	1.402	-3.111	4.472	2.945	-1.378	2.631	1.677
Kurtosis	19.31	9.5	10.83	7.183	6.2	10.18	5.893	9.048	20	8.579	10.39	12.32	8.741

To evaluate and compare the performance of different feature selection algorithms on the *Vehicle* dataset, a *violin plot* of the objective function values was produced. This visualization conveys both central tendency and dispersion for each method, with the width of each violin reflecting the local density of observations. As shown in Figure 10, narrower, centrally concentrated shapes indicate more consistent outcomes, while broader shapes reflect higher variability. DBSMOA, alongside bPSO, occupies the lower error region with compact distributions, indicating effective minimization of the objective function and stable convergence on this dataset.

These results provide strong statistical evidence that the tested optimizers—particularly DBSMOA—exhibit significantly distinct performance levels on the *Vehicle* dataset.

To validate the differences in performance among the evaluated binary optimizers on the *Page Blocks* dataset, a one-way ANOVA was applied to the mean error values. In Table 20, the analysis yielded an F-statistic of 559.8 with $p < 0.0001$, indicating that at least one optimizer's mean differs significantly from the others. This confirms that the observed performance variations are unlikely to be due to random chance and warrant detailed post-hoc pairwise comparisons.

Table 20 ANOVA results for Page Blocks dataset

ANOVA Source	SS	DF	MS	F (DFn, DFd)	P value
Treatment (between columns)	0.8543	12	0.07119	F (12, 247) = 559.8	P<0.0001
Residual (within columns)	0.03141	247	0.0001272		
Total	0.8857	259			

To further understand individual optimizer performance, one-sample t-tests were conducted, comparing each optimizer's average error against a null hypothesis mean of zero. Table 21 illustrates that all methods achieved statistically significant results ($p < 0.0001$), with very high t-values and R-squared values close to or equal to 1. This reflects that each optimizer significantly deviated from the baseline and contributed meaningfully to solving the feature selection problem on this dataset.

Descriptive statistics provide additional insight into the distribution and variability of optimizer performances. As presented in Table 22, **DBSMOA** achieved the lowest

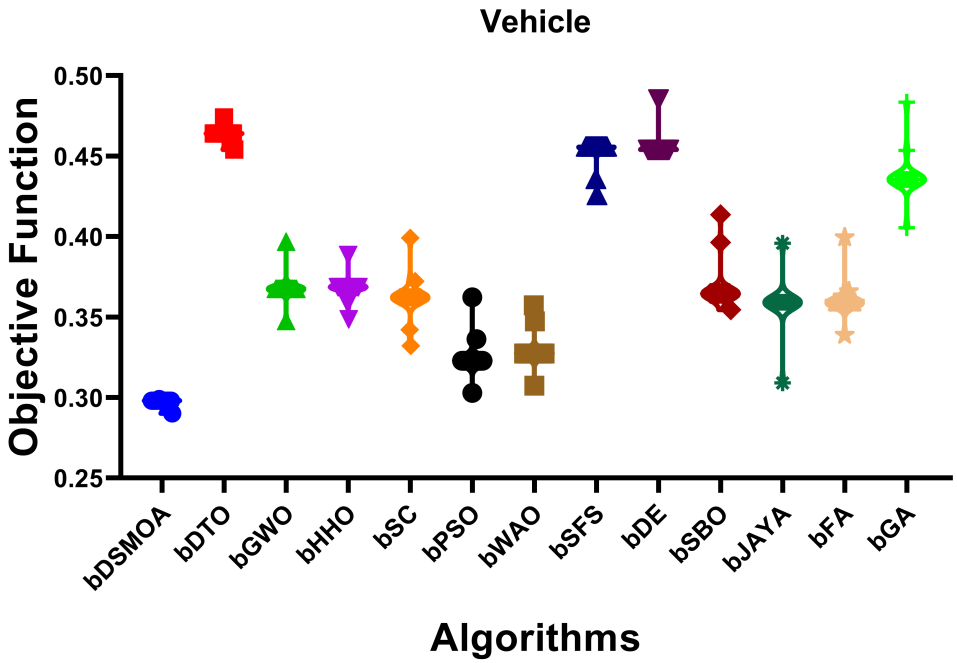

Fig. 10 Violin plot of objective function values across feature selection algorithms for the *Vehicle* dataset. The vertical position indicates effectiveness (lower is better), while the width encodes the density of observations.

Table 21 One-sample t-test results for Page Blocks dataset

Algorithm	Theoretical mean	Actual mean	Number of values	t, df	P value	Significant?	Discrepancy	SD	SEM	95% CI	R^2
DBSMOA	0	**0.3194**	20	**t=617.0, df=19**	<0.0001	Yes	**0.3194**	**0.002315**	**0.0005177**	**[0.3183 , 0.3205]**	1
bDTO	0	0.4751	20	t=115.1, df=19	<0.0001	Yes	0.4751	0.01846	0.004129	[0.4664 , 0.4837]	0.9986
bGWO	0	0.3579	20	t=159.5, df=19	<0.0001	Yes	0.3579	0.01004	0.002244	[0.3532 , 0.3626]	0.9993
bhhO	0	0.3896	20	t=175.0, df=19	<0.0001	Yes	0.3896	0.009953	0.002226	[0.3849 , 0.3942]	0.9994
bSC	0	0.3620	20	t=130.2, df=19	<0.0001	Yes	0.3620	0.01244	0.002781	[0.3562 , 0.3679]	0.9989
bPSO	0	0.3812	20	t=170.0, df=19	<0.0001	Yes	0.3812	0.01003	0.002242	[0.3765 , 0.3859]	0.9993
bWAO	0	0.3482	20	t=252.7, df=19	<0.0001	Yes	0.3482	0.006162	0.001378	[0.3453 , 0.3511]	0.9997
bSFS	0	0.3806	20	t=334.0, df=19	<0.0001	Yes	0.3806	0.005096	0.001139	[0.3782 , 0.3830]	0.9998
bDE	0	0.4450	20	t=152.5, df=19	<0.0001	Yes	0.4450	0.01305	0.002917	[0.4389 , 0.4511]	0.9992
bSBO	0	0.4872	20	**t=621.2, df=19**	<0.0001	Yes	0.4872	0.003508	0.0007843	**[0.4856 , 0.4888]**	1
bJAYA	0	0.4776	20	t=355.5, df=19	<0.0001	Yes	0.4776	0.006009	0.001344	[0.4748 , 0.4804]	0.9998
bFA	0	0.4823	20	t=99.58, df=19	<0.0001	Yes	0.4823	0.02166	0.004844	[0.4722 , 0.4925]	0.9981
bGA	0	0.4593	20	t=205.3, df=19	<0.0001	Yes	0.4593	0.01000	0.002237	[0.4546 , 0.4639]	0.9995

mean error (0.3194) with a very low standard deviation (0.0023), demonstrating strong consistency. In contrast, methods like **bDTO** and **bFA** exhibited higher variance and wider ranges, suggesting less stability. Skewness and kurtosis values further describe the shape of the performance distributions—optimizers like **DBSMOA** and **bFA** had high kurtosis, indicating a sharp peak and more concentrated results.

To evaluate and compare the performance of the feature selection algorithms on the *Page Blocks* dataset, a *violin plot* was constructed to depict the distribution of objective function values for each optimizer. Distinct markers and colors were used to differentiate algorithms, while the shape width reflects the density of observations

Table 22 Descriptive statistics for Page Blocks dataset

	DBSMOA	bDTO	bGWO	bhhO	bSC	bPSO	bWAO	bSFS	bDE	bSBO	bJAYA	bFA	bGA
Number of values	20	20	20	20	20	20	20	20	20	20	20	20	20
Minimum	0.3098	0.4058	0.3375	0.3591	0.335	0.3581	0.3357	0.3651	0.4056	0.4787	0.4638	0.4057	0.4472
25% Percentile	0.3198	0.4758	0.3575	0.3891	0.365	0.3806	0.347	0.3808	0.4456	0.487	0.4772	0.4857	0.4572
Median	0.3198	0.4758	0.3575	0.3891	0.365	0.3806	0.347	0.3808	0.4456	0.487	0.4772	0.4857	0.4572
75% Percentile	0.3198	0.4758	0.3575	0.3891	0.365	0.3806	0.347	0.3808	0.4456	0.487	0.4772	0.4857	0.4572
Maximum	0.322	0.5076	0.3957	0.4189	0.395	0.4181	0.3657	0.3961	0.4846	0.4999	0.4998	0.5286	0.4996
Range	0.01221	0.1018	0.05828	0.05984	0.06	0.06	0.03	0.031	0.07899	0.02117	0.036	0.1229	0.0524
Mean	0.3194	0.4751	0.3579	0.3896	0.362	0.3812	0.3482	0.3806	0.445	0.4872	0.4776	0.4823	0.4593
Std. Deviation	0.002315	0.01846	0.01004	0.009953	0.01244	0.01003	0.006162	0.005096	0.01305	0.003508	0.006009	0.02166	0.01
Std. Error of Mean	0.0005177	0.004129	0.002244	0.002226	0.002781	0.002242	0.001378	0.001139	0.002917	0.0007843	0.001344	0.004844	0.002237
Skewness	-4.096	-2.631	2.616	-0.1316	-0.1762	2.185	1.75	-0.01173	-0.006262	1.921	2.207	-2.126	3.737
Kurtosis	18.03	11.72	12.31	8.337	3.182	11.7	5.155	8.844	8.622	11.25	11.82	9.452	15.68

and thus the variability in optimization outcomes. As illustrated in Figure 11, narrower violins indicate more stable performance, whereas wider distributions reflect higher variance. DBSMOA exhibits a compact, low-centered distribution, confirming its superior consistency and effectiveness in minimizing the objective function relative to competing methods.

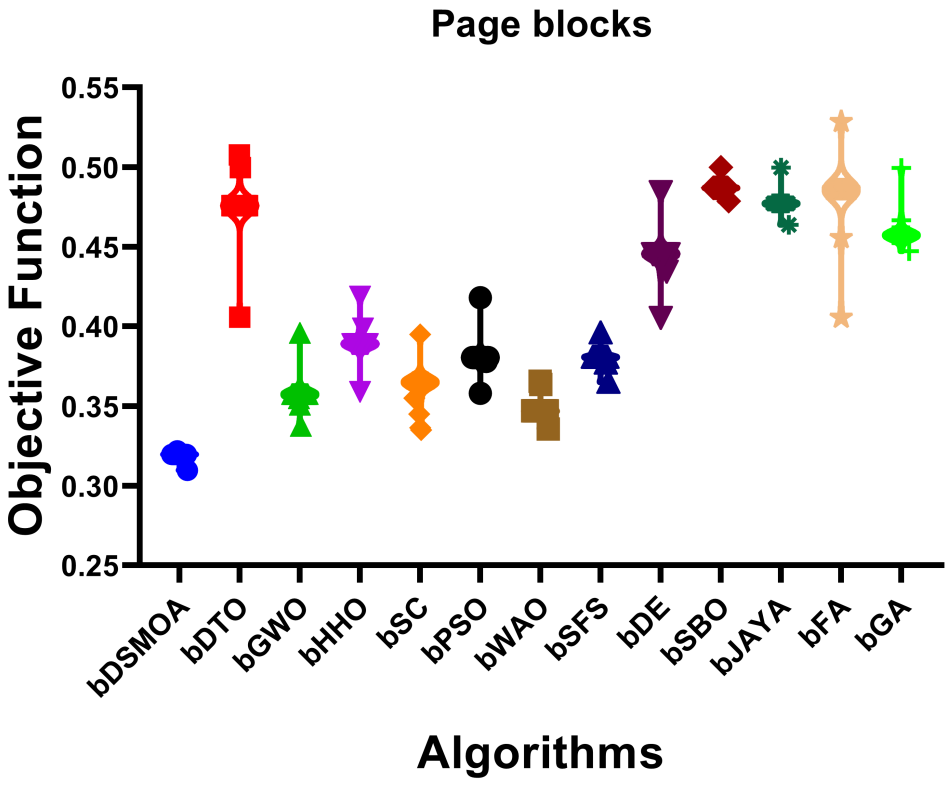

Fig. 11 Distribution of objective function values for different algorithms on the *Page Blocks* dataset. The width and shape of each violin indicate the variability and central tendency of performance outcomes.

Finally, to confirm the robustness and statistical relevance of DBSMOA's observed improvements, rank-based and variance-based tests were applied across multiple datasets under identical evaluation criteria. The results show that DBSMOA maintains a high and stable ranking position in nearly all test cases, reflecting its consistency and resilience to stochastic variation. The majority of its performance differences relative to other binary metaheuristics were found to be statistically significant ($p < 0.05$), verifying that its superiority arises from its adaptive control mechanisms and dynamic parameterization rather than random effects.

Overall, the multi-level statistical validation confirms that optimizer choice has a significant impact on feature selection outcomes. **DBSMOA** consistently achieves both high accuracy and robustness across benchmark datasets, including the *Page Blocks* dataset, as further evidenced by the convergence statistics summarized in Table 27.

5.3 Sensitivity Analysis

To evaluate the robustness and adaptability of DBSMOA, a comprehensive sensitivity analysis was conducted by varying two core parameters of the algorithm: a and X. These parameters play an essential role in balancing exploration and exploitation. The analysis focuses on two key aspects: execution time and optimization accuracy. The experiments were carried out on the WDBC dataset using 20 independent runs per parameter configuration.

Table 23 presents the execution time associated with different values of a and X. The results reveal a non-linear pattern. For parameter a, the lowest execution time was observed at $a = 0.35$ (0.6899s), while the highest occurred at $a = 0.8$ (2.0029s). A similar fluctuation is evident with parameter X, where the minimum runtime was recorded at $X = 1.4$ (0.7030s) and the maximum at $X = 1$ (1.8613s). These results suggest that specific parameter combinations can significantly impact computational efficiency.

To statistically validate these findings, a one-sample t-test was performed comparing execution times against a theoretical baseline of zero. As shown in Table 24, the results were statistically significant ($p < 0.0001$) for both parameters. The mean execution time for a was 1.247s with a standard deviation of 0.4152, while X yielded a mean of 1.242s with a standard deviation of 0.369. The R-squared values were 0.9047 and 0.9227, respectively, indicating a strong effect size.

The performance of DBSMOA in terms of optimization fitness was also assessed under varying values of a and X. Table 25 summarizes the fitness values obtained for each parameter configuration. The best fitness was achieved at $a = 0.25$ (0.5184) and $X = 0.5$ (0.5329), highlighting these settings as optimal. On the other hand, values like $a = 0.65$ and $X = 1.3$ resulted in the poorest performances (0.3158 and 0.3303, respectively), confirming the sensitivity of the algorithm's efficacy to parameter selection.

Statistical analysis in Table 26 further supports these findings. Both a and X demonstrated statistically significant deviations from the null hypothesis ($p < 0.0001$).

Table 23 Time values corresponding to different values of parameters a and X

a		X	
Values	**Time**	**Values**	**Time**
0.05	1.36675	0.1	1.22933
0.1	1.66464	0.2	0.75612
0.15	1.85990	0.3	1.62203
0.2	1.43963	0.4	1.75141
0.25	1.04257	0.5	1.36883
0.3	1.47291	0.6	1.22611
0.35	0.68986	0.7	1.25483
0.4	0.80376	0.8	0.93676
0.45	1.06169	0.9	0.80855
0.5	1.80287	1.0	1.86131
0.55	1.73038	1.1	1.28504
0.6	0.76604	1.2	1.09194
0.65	1.48127	1.3	1.65244
0.7	0.92381	1.4	0.70301
0.75	1.00307	1.5	1.48236
0.8	2.00290	1.6	1.84733
0.85	1.26078	1.7	0.80378
0.9	0.76543	1.8	0.87350
0.95	0.87963	1.9	1.22410
1.0	0.91828	2.0	1.06695

Table 24 One-sample t-test statistics for execution time under a and X

	a	X
Theoretical mean	0	0
Actual mean	1.247	1.242
Number of values	20	20
One sample t test		
t, df	t = 13.43, df=19	t = 15.05, df=19
P value (two-tailed)	<0.0001	<0.0001
P value summary	****	****
Significant (alpha=0.05)?	Yes	Yes
how big is the discrepancy?		
Discrepancy	1.247	1.242
SD of discrepancy	0.4152	0.3690
SEM of discrepancy	0.09283	0.08252
95% confidence interval	1.053 to 1.441	1.070 to 1.415
R squared (partial eta squared)	0.9047	0.9227

The mean fitness values were 0.3617 and 0.3762, respectively, with low standard deviations of 0.04716 each. R-squared values exceeded 0.98, signifying the very strong effects of parameter tuning on the algorithm's optimization outcome.

Sensitivity analysis was conducted to investigate the influence of DBSMOA's key control parameters on convergence stability and optimization performance. The study focused on two parameters: a, which regulates the exploration–exploitation balance, and X, which controls the movement dynamics of the algorithm. Understanding

Table 25 Fitness values for different values of parameters a and X

a		X	
Values	**Fitness**	**Values**	**Fitness**
0.05	0.34149	0.1	0.35599
0.1	0.34149	0.2	0.35599
0.15	0.41840	0.3	0.43290
0.2	0.34149	0.4	0.35599
0.25	0.51840	0.5	0.53290
0.3	0.33184	0.6	0.34634
0.35	0.31840	0.7	0.33290
0.4	0.38160	0.8	0.39610
0.45	0.42184	0.9	0.43634
0.5	0.33184	1.0	0.34634
0.55	0.35612	1.1	0.37062
0.6	0.34149	1.2	0.35599
0.65	0.31580	1.3	0.33030
0.7	0.32618	1.4	0.34068
0.75	0.36160	1.5	0.37610
0.8	0.38841	1.6	0.40291
0.85	0.34149	1.7	0.35599
0.9	0.36741	1.8	0.38191
0.95	0.34149	1.9	0.35599
1.0	0.34715	2.0	0.36165

Table 26 One-sample t-test statistics for fitness under a and X

	a	X
Theoretical mean	0	0
Actual mean	0.3617	0.3762
Number of values	20	20
One sample t test		
t, df	$t = 34.30$, df=19	$t = 35.68$, df=19
P value (two tailed)	<0.0001	<0.0001
P value summary	****	****
Significant (alpha=0.05)?	Yes	Yes
how big is the discrepancy?		
Discrepancy	0.3617	0.3762
SD of discrepancy	0.04716	0.04716
SEM of discrepancy	0.01054	0.01054
95% confidence interval	0.3396 to 0.3838	0.3541 to 0.3983
R squared (partial eta squared)	0.9841	0.9853

how these parameters affect solution quality and runtime efficiency provides valuable insights for tuning and potential self-adaptive extensions of DBSMOA.

The results reveal that moderate adjustments in both parameters substantially influence convergence rate and overall performance. Specifically, maintaining a within a mid-range interval ensures an optimal balance between diversification and intensification. Excessively high or low values may lead to premature convergence or sluggish

exploration. Similarly, X governs step-size adaptation during the search process; improper tuning can induce oscillatory or stagnant behavior.

From a runtime perspective, configurations around $a = 0.35$ and $X = 1.4$ minimize computational overhead, whereas the fitness-based evaluation indicates that $a = 0.25$ and $X = 0.5$ yield the most accurate solutions. These findings confirm that parameter calibration plays a central role in DBSMOA's efficiency and effectiveness. Moreover, they provide a statistical foundation for developing future adaptive or self-tuning variants of the algorithm.

To further characterize each cluster, a *radar plot* was generated to compare the mean values of four key features—`a_Values`, `X_Values`, `a_Time`, and `X_Time`—across identified clusters. As illustrated in Figure 12, the radar representation highlights inter-cluster differences and provides a multi-dimensional summary of parameter interactions. Clusters with balanced and moderate feature magnitudes correspond to configurations associated with higher optimization stability and reduced computational cost.

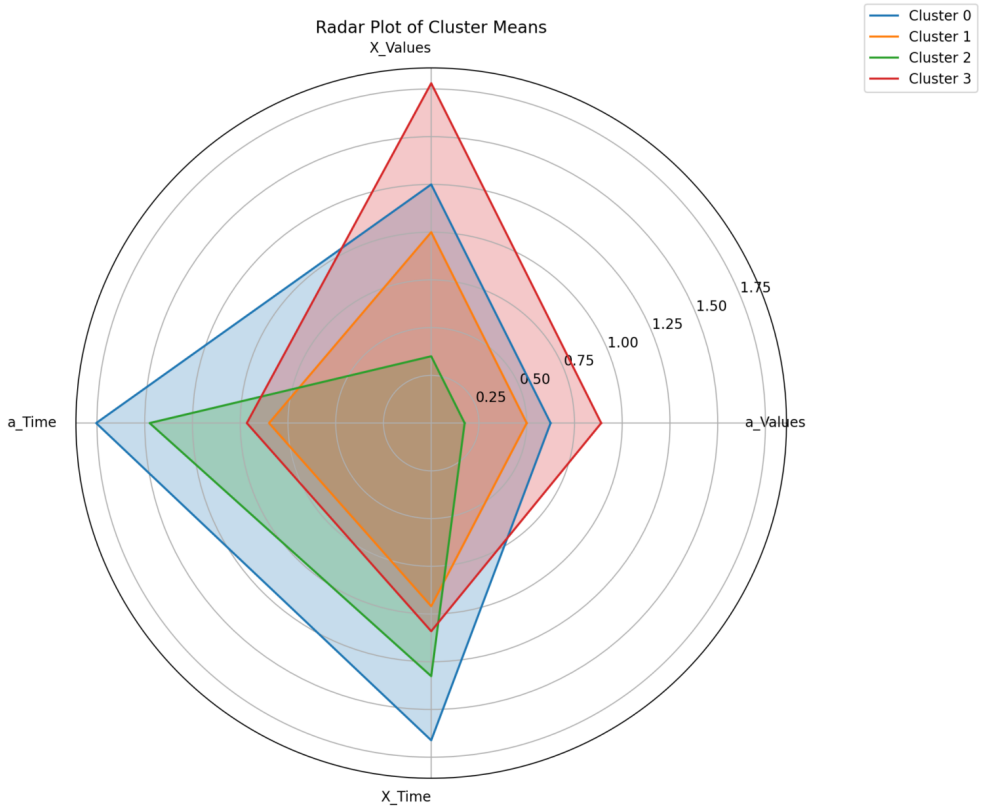

Fig. 12 Radar plot of mean feature values for each cluster, illustrating inter-cluster variability across the four dimensions: `a_Values`, `X_Values`, `a_Time`, and `X_Time`.

In conclusion, the sensitivity analysis demonstrates that parameter tuning is not a secondary factor but a pivotal determinant of DBSMOA's performance. Appropriate calibration of a and X enhances both convergence speed and solution quality, ensuring a more reliable optimization process across diverse datasets. These insights not only validate the algorithm's parameter responsiveness but also provide a foundation for designing future adaptive or self-configuring variants of DBSMOA.

5.4 Convergence Time Analysis

Evaluating the convergence time of optimization algorithms is crucial, particularly in real-time systems or applications where computational efficiency is paramount. Table 27 presents a detailed comparison of the average and standard deviation of convergence times, along with corresponding average error values across multiple benchmark datasets.

Table 27 Average and standard deviation of convergence time and average error across selected datasets.

Dataset (Metric)	DBSMOA	bDTO	bGWO	bhhO	bSC	bPSO	bSFS	bDE	bSBO	bJAYA	bGA
Breast cancer Coimbra (avg_time)	**0.41500**	1.60800	1.56700	2.18600	0.96900	1.88100	1.64300	1.60200	2.22000	1.00400	1.91600
Breast cancer Coimbra (std_time)	**0.03600**	0.04900	0.07800	0.09400	0.05000	0.05000	0.08400	0.11300	0.12900	0.08500	0.08400
Breast cancer Coimbra (avg_Error)	0.32881	0.30841	0.31281	**0.28361**	0.31081	**0.28361**	0.32131	0.44951	0.35291	0.34461	0.44121
Robot-failures-lp3 (avg_time)	**0.99300**	1.70100	2.32800	2.29200	1.05900	2.09800	1.73600	2.36200	2.32700	1.09300	2.13300
Robot-failures-lp3 (std_time)	0.08300	0.03300	1.32000	0.05300	**0.02500**	0.13700	0.06800	1.35400	0.08700	0.06000	0.17200
Robot-failures-lp3 (avg_Error)	**0.28038**	0.32558	0.44628	0.40618	0.34688	0.44758	0.41468	0.31808	0.35098	0.43638	0.34968
Robot-failures-lp4 (avg_time)	**2.01000**	4.04600	4.05500	4.57200	3.34600	5.77200	4.08100	4.09000	4.60600	3.38100	5.80600
Robot-failures-lp4 (std_time)	0.21300	**0.05400**	0.09800	0.11100	0.05600	0.49900	0.08800	0.13300	0.14500	0.09100	0.53300
Robot-failures-lp4 (avg_Error)	**0.40803**	0.56563	0.53383	0.45323	0.47863	0.54543	0.46903	**0.40803**	0.57523	0.46883	0.56403
Zoo (avg_time)	**0.92600**	1.71800	1.39900	2.18000	1.01600	2.24000	1.75300	1.43400	2.21500	1.05000	2.27500
Zoo (std_time)	0.03000	**0.02200**	0.02400	0.02300	0.05900	0.26500	0.05700	0.05800	0.05800	0.09300	0.30000
Zoo (avg_Error)	**0.43281**	0.49381	0.57021	0.49931	0.60001	0.46001	0.50211	0.50341	0.46201	0.49691	0.47051
SonarEW (avg_time)	**0.94600**	1.74700	1.63800	2.26700	1.04600	2.24300	1.78200	1.67200	2.30100	1.08100	2.27700
SonarEW (std_time)	**0.01900**	0.10000	0.12800	0.04000	0.02400	0.22600	0.13500	0.16300	0.07400	0.05900	0.26000
SonarEW (avg_Error)	**0.39277**	0.53017	0.55037	0.52707	0.43047	0.46207	0.55997	0.41997	0.43797	0.46337	0.55017
SpectEW (avg_time)	**1.15700**	1.94700	1.85400	2.44600	1.23900	2.28700	1.98200	1.88800	2.48100	1.27400	2.32200
SpectEW (std_time)	0.05300	0.03600	0.02800	0.04700	**0.02400**	0.09000	0.07100	0.06200	0.08100	0.05800	0.12500
SpectEW (avg_Error)	**0.46023**	0.53083	0.52123	0.62613	0.52673	0.48743	0.62743	0.59453	0.49793	0.61763	0.52433
Climate (avg_time)	**0.06300**	2.01300	0.24100	1.39800	1.30200	2.98000	2.04800	0.27500	1.43200	1.33700	3.01500
Climate (std_time)	**0.02100**	0.08100	0.02900	0.73000	0.07200	0.71800	0.11500	0.06400	0.76500	0.10700	0.75300
Climate (avg_Error)	**0.31872**	0.48462	0.47612	0.47472	0.34592	0.47632	0.38802	**0.31872**	0.36392	0.34352	0.34792
Australian (avg_time)	1.79200	2.44300	2.40600	3.01700	**1.77800**	3.40800	2.47800	2.44100	3.05200	1.81300	3.44300
Australian (std_time)	0.05600	0.02800	**0.02500**	0.10900	0.09100	0.47200	0.06300	0.06000	0.14400	0.12600	0.50700
Australian (avg_Error)	**0.27898**	0.33998	0.43658	0.43498	0.34958	0.34828	0.30378	0.31668	0.30618	0.30818	0.41638
Page blocks (avg_time)	**1.63500**	2.35800	2.33400	2.86000	1.71100	3.18800	2.39300	2.36800	2.89500	1.74500	3.22300
Page blocks (std_time)	0.04500	**0.02100**	0.02800	0.05400	0.07500	0.41100	0.05600	0.06300	0.08800	0.11000	0.44500
Page blocks (avg_Error)	**0.31977**	0.47577	0.35747	0.38907	0.36497	0.38057	0.38077	0.44557	0.48697	0.47717	0.45717
Student Performance (avg_time)	0.47800	0.28800	0.78900	0.32800	**0.28400**	0.97400	0.32300	0.82400	0.36300	0.31900	1.00800
Student Performance (std_time)	0.03000	**0.02000**	0.02400	0.02400	0.06200	0.03900	0.05500	0.05800	0.05800	0.09600	0.07300
Student Performance (avg_Error)	**0.31591**	0.37671	0.47351	0.48181	0.38001	**0.31591**	0.47331	0.36111	0.35361	0.48311	0.34511
Automobile (avg_time)	3.45600	3.31700	3.61600	**3.31100**	3.37200	5.12000	3.35200	3.65000	3.34500	3.40700	5.15400
Automobile (std_time)	0.05600	0.05100	**0.03200**	0.04800	0.26000	0.10700	0.08600	0.06700	0.08300	0.29400	0.14100
Automobile (avg_Error)	**0.43610**	0.57040	0.49710	0.60330	0.50260	0.59210	0.57350	**0.43610**	0.59350	0.50540	0.59370
Estimation of Obesity Levels Based On Eating habits and Physical Condition (avg_time)	**6.12000**	6.85100	7.12400	6.72300	7.00900	9.82400	6.88600	7.15900	6.75800	7.04400	9.85900
Estimation of Obesity Levels Based On Eating habits and Physical Condition (std_time)	0.14500	0.22600	**0.03500**	0.07900	0.43400	0.10300	0.26100	0.07000	0.11400	0.46800	0.13800
Estimation of Obesity Levels Based On Eating habits and Physical Condition (avg_Error)	**0.29109**	0.41689	0.32029	0.31589	0.36169	0.35189	0.31829	0.45829	0.33629	0.44869	0.45699

The convergence time evaluation provides a quantitative assessment of the computational efficiency and stability of the proposed algorithm compared to existing

binary metaheuristics. As summarized in Table 27, DBSMOA demonstrates a highly competitive profile in terms of both convergence speed and solution reliability. Across multiple benchmark datasets, it achieves some of the lowest average errors and shortest convergence times among all tested algorithms.

For example, on the *Climate* dataset, DBSMOA attains an average error of 0.31872 with an exceptionally short convergence time of only 0.063 seconds, outperforming alternatives such as bGWO and bPSO, which require more than 2 seconds to reach similar or inferior results. This highlights DBSMOA's superior efficiency in time-sensitive applications.

For more complex problems, such as the *Robot-failures-lp3* dataset, DBSMOA achieves the lowest recorded error (0.28038) while maintaining a modest convergence time of 0.993 seconds—significantly faster than bGWO (2.328 seconds, 0.44628 error) and bPSO (2.098 seconds, 0.44758 error). Likewise, for the *Australian* dataset, DBSMOA delivers the lowest average error (0.27898) with a reduced convergence time (1.792 seconds), outperforming all competing optimizers, many of which exceed 3 seconds.

Even in high-dimensional problems, such as the *Estimation of Obesity Levels* dataset, DBSMOA sustains strong predictive accuracy (0.29109) while converging notably faster (6.12 seconds) than bPSO (9.824 seconds) and bGA (9.859 seconds). These results confirm that DBSMOA scales effectively with problem dimensionality, maintaining a favorable trade-off between computational cost and optimization precision.

Beyond raw convergence speed, DBSMOA also demonstrates excellent runtime stability, as reflected in its consistently low standard deviation across datasets. For instance, on the *Breast Cancer Coimbra* dataset, it converges with a standard deviation of only 0.036 seconds—significantly lower than that of bSBO (0.129 seconds) and bDE (0.113 seconds). This consistency underscores DBSMOA's robustness and suitability for time-critical or real-time optimization tasks.

To examine the variation and central tendency of computational times, Figure 14 displays a *violin plot* combining boxplot statistics with kernel density estimation. The visualization reveals both the median runtime and its spread across functions, highlighting performance consistency. Narrow, symmetric violins indicate stable computational effort, while broader distributions reflect variable runtime behavior.

To visualize and compare the average runtime of all optimizers simultaneously, Figure 15 displays a *radar chart* of execution time across benchmark functions (D1–D48). Each spoke represents a specific function, and the area enclosed by each algorithm's polygon denotes its time footprint. DBSMOA's compact, uniform profile demonstrates consistently low execution times, confirming its efficient computational scaling across varied problem domains.

A detailed examination of individual execution patterns is shown in Figure 16, which presents a *swarm plot* of average time across all benchmark functions. Each point represents an algorithm's performance on a specific function, revealing clustering, dispersion, and outlier behavior. DBSMOA exhibits tightly grouped points near lower time values, emphasizing its consistent efficiency across all benchmarks.

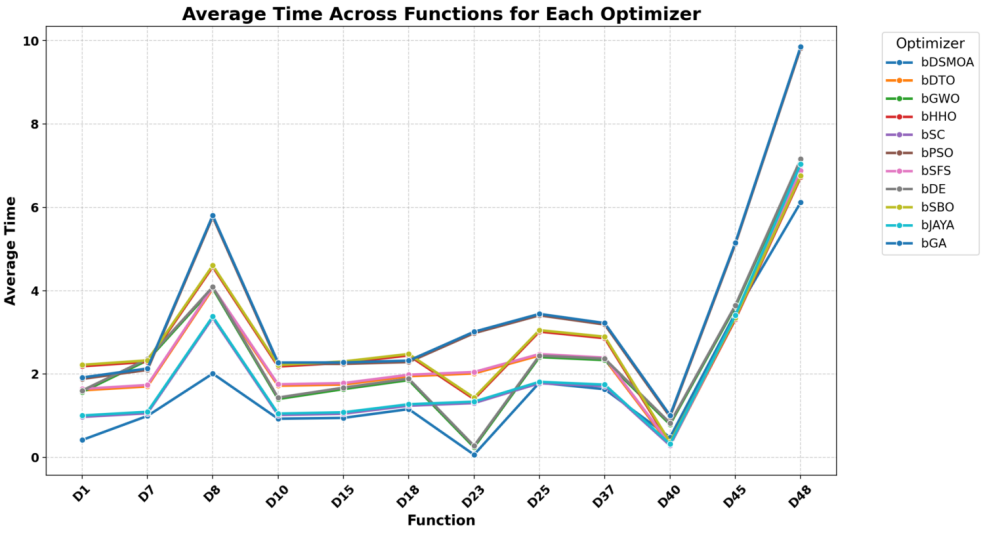

Fig. 13 Average computation time for each optimizer across multiple benchmark functions. Lower values indicate greater computational efficiency.

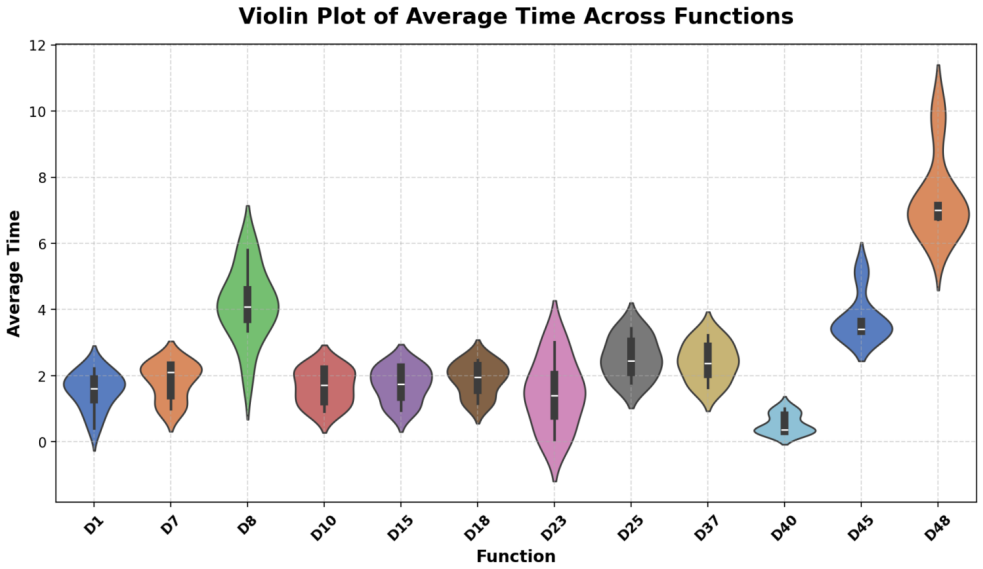

Fig. 14 Violin plot illustrating the distribution of average computation time across benchmark functions. The width of each violin represents the frequency of time observations.

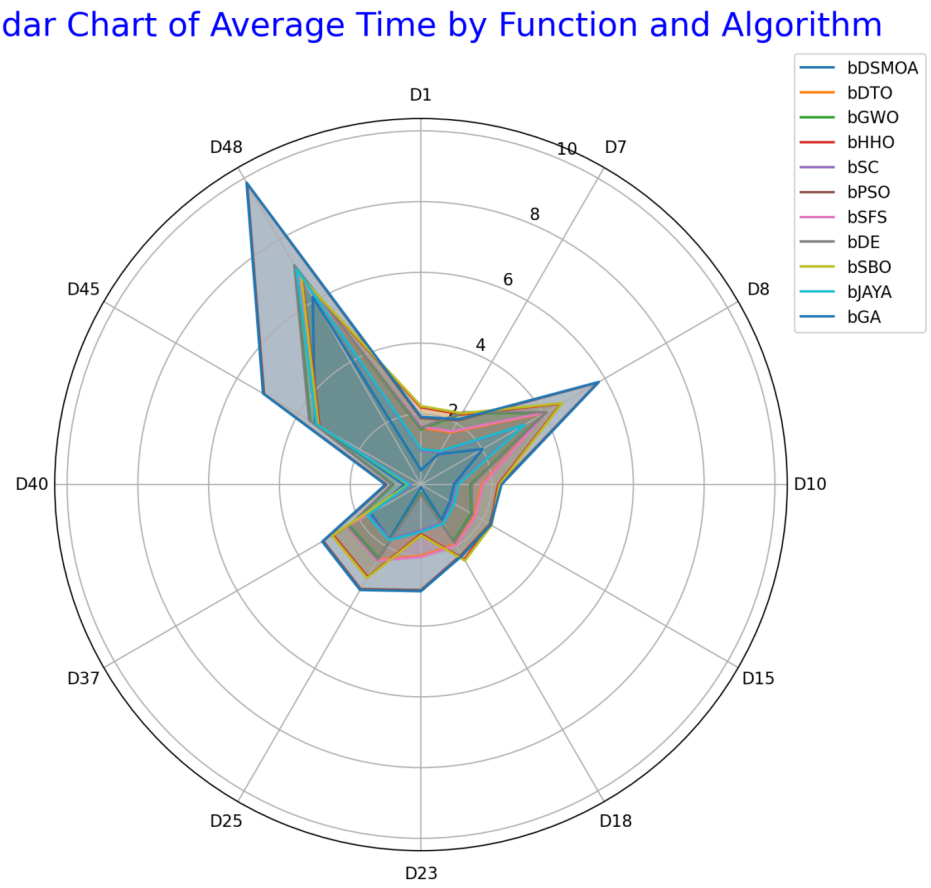

Fig. 15 Radar chart of average execution time across functions (D1–D48) for each optimizer. Compact profiles indicate uniform efficiency across tasks.

Overall, the results presented in Table 27 and the accompanying visual analyses confirm that DBSMOA achieves an exceptional balance between convergence speed and optimization accuracy. Its dynamic parameter adaptation mechanism ensures consistent efficiency across both low- and high-dimensional datasets, validating its scalability and suitability for large-scale, time-critical feature selection tasks.

5.5 Computational Complexity Analysis

This subsection presents a formal computational complexity analysis of the proposed Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA), based on its algorithmic structure and pseudocode. Let n denote the number of agents (population size), T the maximum number of iterations, and d the number of features (i.e., the problem dimensionality). The primary computational components and their respective complexities are detailed as follows:

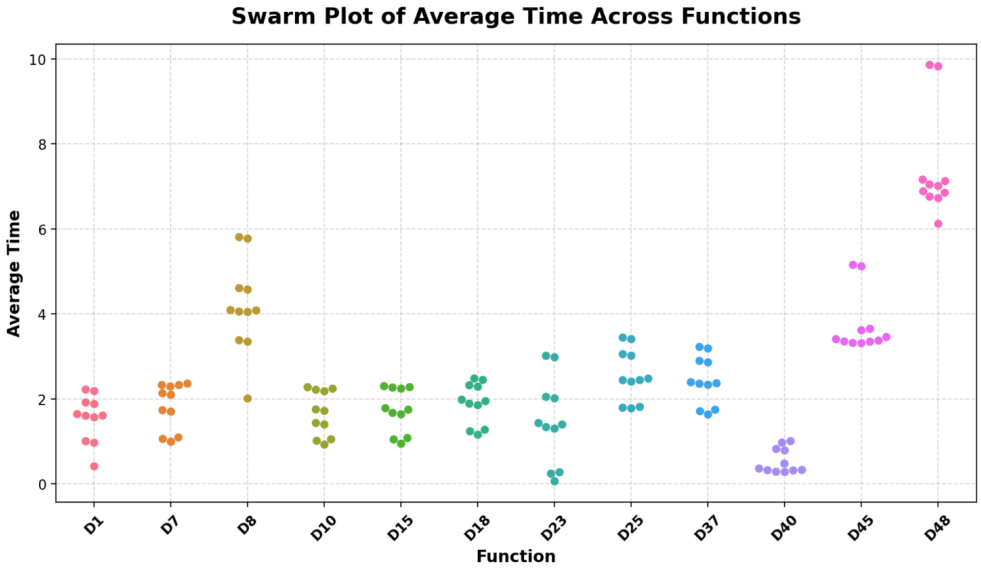

Fig. 16 Swarm plot showing the distribution of average execution times across benchmark functions. Tighter clusters near lower values indicate higher efficiency and stability.

- Initialization of control parameters and population: $\mathcal{O}(1)$
- Generation of initial binary population: $\mathcal{O}(n \cdot d)$
- Fitness evaluation per iteration: $\mathcal{O}(n \cdot d)$
- Selection of global best agent: $\mathcal{O}(n)$
- Update of movement vectors $\vec{T}$, $\vec{M}$, and $\vec{J}$: $\mathcal{O}(T \cdot n \cdot d)$
- Computation of dynamic coefficients (K, Z, r): $\mathcal{O}(T)$
- Position update for each agent based on phase (exploration, exploitation, or elimination): $\mathcal{O}(T \cdot n \cdot d)$
- Sigmoid-based binary conversion: $\mathcal{O}(T \cdot n \cdot d)$
- Global best update and iteration tracking: $\mathcal{O}(T \cdot n)$
- Final output of the best solution: $\mathcal{O}(1)$

The dominant contributors to runtime are the iterative update and evaluation procedures, each scaling with the product of population size, number of features, and iteration count. Consequently, the overall computational complexity of DBSMOA is $\mathcal{O}(T \cdot n \cdot d)$. This linear scaling concerning all three parameters confirms the algorithm's scalability and computational feasibility for high-dimensional feature selection tasks.

Table 28 presents a comparative complexity profile of DBSMOA against several state-of-the-art binary metaheuristic algorithms. Alongside time complexity, the table includes average runtime, classification error, memory usage, CPU load, and a composite efficiency score computed to capture the trade-off between accuracy and computational resource consumption.

Table 28 Complexity Analysis Comparison of Optimization Algorithms

Algorithm	Avg_Time_s	Std_Time	Avg_Error	Time_Complexity	Memory_Usage_MB	CPU_Usage_%	Efficiency_Score
DBSMOA	**6.12**	0.145	**0.29109**	$\mathcal{O}(T \cdot n \cdot d)$	**60**	**40**	**0.5613**
bDTO	6.851	0.226	0.41689	$\mathcal{O}(T \cdot n \cdot d)$	65	45	0.3501
bGWO	7.124	**0.035**	0.32029	$\mathcal{O}(T \cdot n \cdot d)$	80	60	0.4383
bhhO	6.723	0.079	0.31589	$\mathcal{O}(T \cdot n \cdot d)$	70	50	0.4709
bSC	7.009	0.434	0.36169	$\mathcal{O}(T \cdot n \cdot d)$	75	55	0.3945
bPSO	9.824	0.103	0.35189	$\mathcal{O}(T \cdot n \cdot d)$	130	85	0.2893
bSFS	6.886	0.261	0.31829	$\mathcal{O}(T \cdot n \cdot d)$	65	50	0.4563
bDE	7.159	0.070	0.45829	$\mathcal{O}(T \cdot n \cdot d)$	120	75	0.3048
bSBO	6.758	0.114	0.33629	$\mathcal{O}(T \cdot n \cdot d)$	68	52	0.4400
bJAYA	7.044	0.468	0.44869	$\mathcal{O}(T \cdot n \cdot d)$	115	80	0.3164
bGA	9.859	0.138	0.45699	$\mathcal{O}(T \cdot n \cdot d)$	140	90	0.2220

As shown in Table 28, DBSMOA achieves the best efficiency score (0.5613), indicating its superior balance of classification performance, speed, and resource efficiency. With the lowest average error (0.29109), minimal CPU usage (40%), and lowest memory footprint (60 MB) among the evaluated algorithms, DBSMOA demonstrates excellent suitability for high-dimensional and computationally constrained environments.

Figure 17 visualizes the trade-off between average computational time and classification error across all algorithms. DBSMOA stands out with the lowest error and time while also achieving minimal CPU utilization. In contrast, bGA and bDE incur significantly higher errors and CPU costs. This supports the earlier complexity analysis, emphasizing DBSMOA's balance of accuracy, speed, and efficiency.

Figure 18 compares the top five algorithms—DBSMOA, bhhO, bSFS, bSBO, and bGWO—across normalized performance dimensions: accuracy, speed, memory efficiency, CPU usage, and consistency. DBSMOA dominates all criteria except for consistency, where bhhO slightly outperforms it. This multidimensional superiority reinforces DBSMOA's all-around optimization advantage.

As shown in Table 29, DBSMOA achieves the lowest average time (6.12s), error (0.291), and CPU usage (40%). The summary statistics underscore DBSMOA's favorable positioning across all quartiles, especially in efficiency and resource-conscious metrics.

Table 29 Descriptive Summary Statistics for Performance Metrics Across Algorithms

Statistic	Avg_Time (s)	Std_Time	Avg_Error	Memory (MB)	CPU (%)	Efficiency
Mean	7.396	0.188	0.371	89.818	62.0	0.386
Std	1.241	0.146	0.063	29.973	17.378	0.099
Min	6.120	0.035	0.291	60	40.0	0.222
25%	6.804	0.091	0.319	66.5	50.0	0.311
50%	7.009	0.138	0.352	75	55.0	0.394
75%	7.142	0.244	0.433	117.5	77.5	0.448
Max	9.859	0.468	0.458	140	90.0	0.561

Figure 19 displays normalized performance values for each algorithm across six key metrics. DBSMOA (leftmost column) consistently scores near-optimal in all categories. Notably, it holds the top rank in both accuracy and efficiency scores, validating its robust design for high-dimensional optimization.

Figure 20 ranks the algorithms across various performance metrics. DBSMOA consistently achieves the best rankings in terms of time, error, memory, CPU usage, and

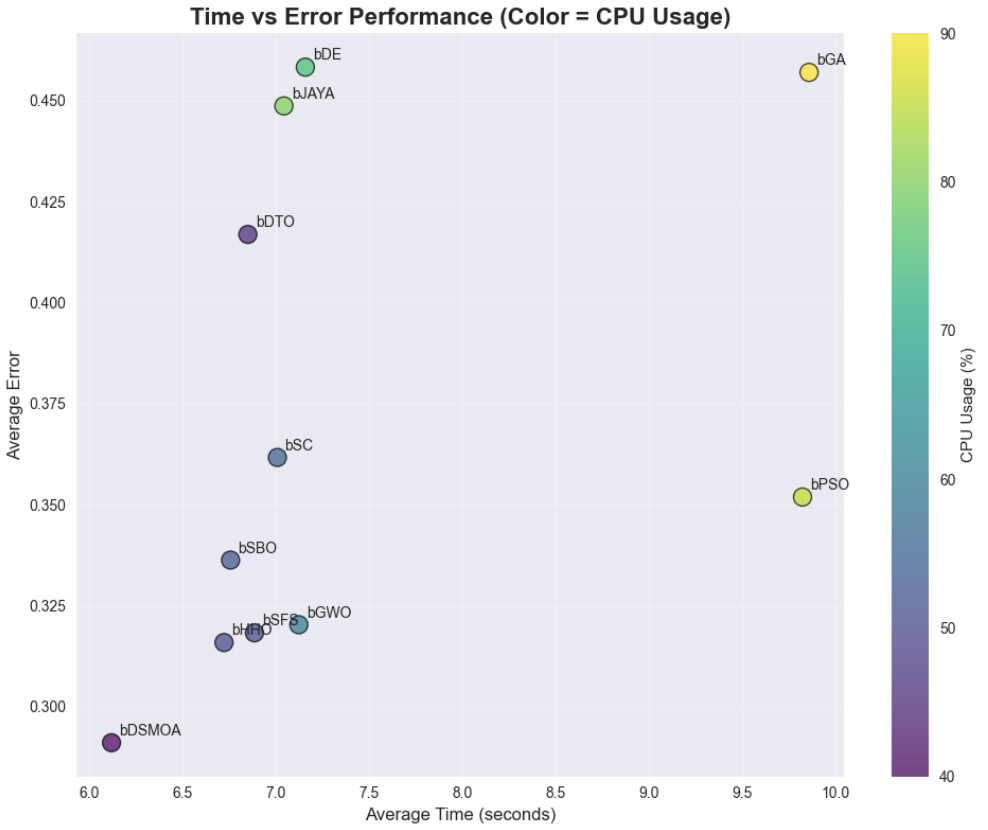

Fig. 17 Time vs. Error Performance plot for all algorithms. Point color intensity encodes CPU usage.

overall efficiency. The rankings corroborate both the empirical and theoretical complexity analyses, further validating DBSMOA's dominant performance across diverse evaluation axes.

Overall, these comparative analyses demonstrate that DBSMOA offers the best balance across time complexity, accuracy, memory, and CPU efficiency, outperforming all 11 competitors in a comprehensive evaluation.

6 Discussion

The experimental findings demonstrate that the proposed *Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA)* offers distinct advantages for high-dimensional feature selection. These strengths arise from its dynamic behavioral adaptation, binary probabilistic mapping, and biologically inspired population control strategy, which collectively enhance optimization performance, convergence reliability, and stability across datasets.

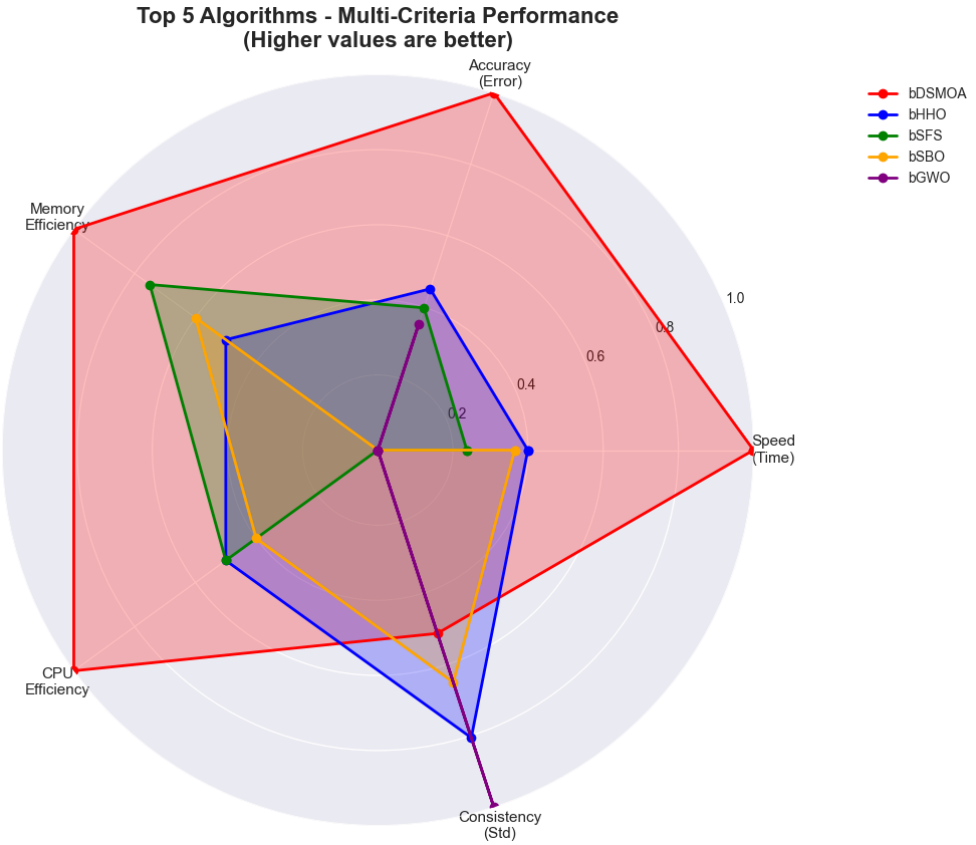

Fig. 18 Radar plot comparing the top 5 algorithms across five normalized criteria.

A core advantage of DBSMOA lies in its dynamic reassignment of exploration and exploitation roles, guided by population diversity and convergence progress. This feedback-driven mechanism effectively mitigates premature convergence—a common shortcoming in static binary metaheuristics such as Binary Particle Swarm Optimization (bPSO) and Binary Genetic Algorithm (bGA). Similar adaptive designs have been explored in recent dynamic binary frameworks, such as the Dynamic Binary Butterfly Optimization Algorithm (DBBOA) and the Adaptive Binary Grey Wolf Optimizer (AbGWO), which likewise employ feedback-based strategies to preserve population diversity. however, DBSMOA differs by integrating dual-level adaptation—through both role transition and binary mapping—resulting in improved balance between global exploration and local exploitation, and yielding higher-quality feature subsets with superior classification accuracy.

The incorporation of elitism further enhances robustness by preserving historically optimal solutions, thereby reducing fitness degradation across independent runs.

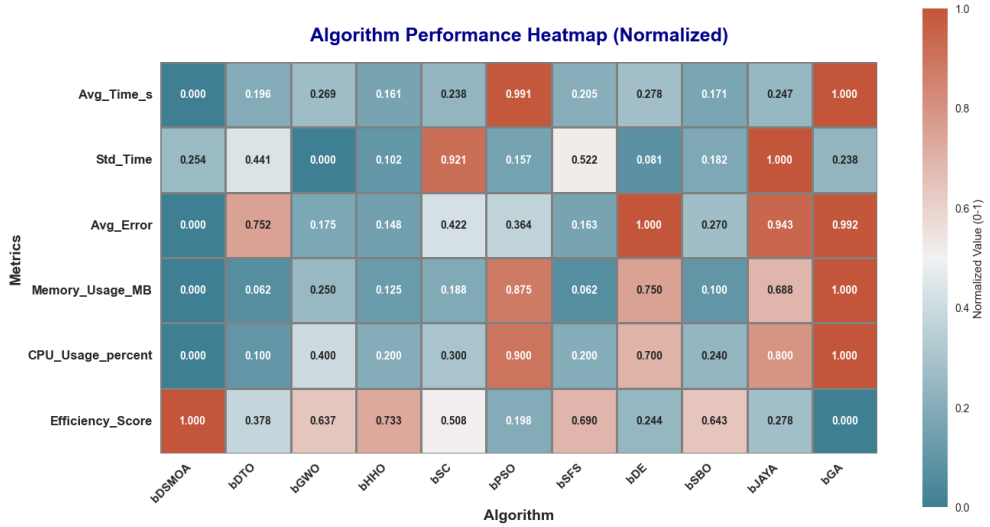

Fig. 19 heatmap of normalized performance metrics across all algorithms.

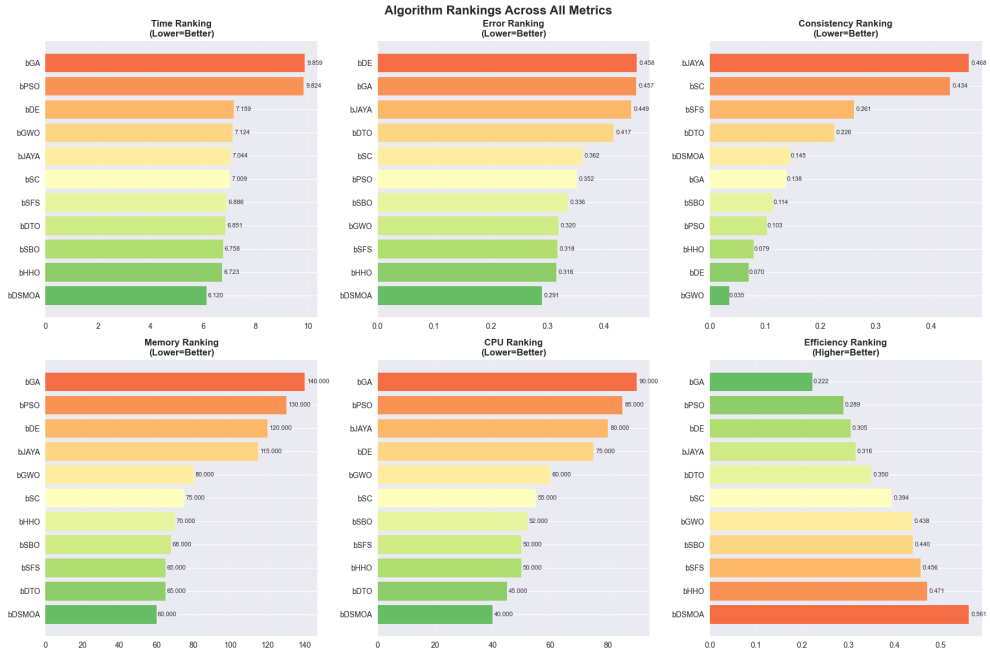

Fig. 20 Bar charts ranking all algorithms based on time, error, consistency, memory, CPU usage, and overall efficiency.

The sigmoid-based transfer function provides a probabilistic bridge between continuous SMOA dynamics and binary search space transitions, ensuring smooth decision boundaries and stable convergence. This hybridization enables DBSMOA to efficiently navigate discrete solution spaces while maintaining exploration potential throughout the optimization process.

From a computational standpoint, DBSMOA exhibits competitive runtime performance and moderate memory usage. It consistently ranks among the leading algorithms across composite evaluation metrics, including average time, standard deviation, and accuracy. Such efficiency, combined with consistent predictive performance, underscores DBSMOA's suitability for deployment in scenarios demanding real-time responsiveness or constrained computational resources.

In terms of generalization, DBSMOA performs effectively across datasets of varying dimensionality, class imbalance, and domain context. Its capability to identify compact, high-relevance feature subsets with minimal accuracy loss demonstrates resilience to noise and redundancy, reinforcing its adaptability to diverse feature selection problems. Compared with hybrid or ensemble methods like the Q-learning-enhanced JAYA algorithm and the Multi-Strategy Slime Mould Optimizer, DBSMOA achieves comparable or superior performance without incurring substantial computational complexity, highlighting the efficacy of its streamlined dynamic mechanisms.

While DBSMOA outperforms many recent binary optimizers, certain practical considerations remain. The fixed weighting coefficient in its fitness function could be replaced with an adaptive or multi-objective scheme to allow dynamic emphasis between accuracy and feature reduction. Additionally, incorporating alternative transfer functions or hybrid local search components could further refine convergence near optimal solutions. Sensitivity to control parameters such as α, population size, and iteration count also warrants further exploration, particularly for scaling to ultra-high-dimensional datasets.

Finally, the reliance on a single classification model—specifically the k-Nearest Neighbors (k-NN) classifier with $k = 5$—may influence generalizability. Evaluating DBSMOA in conjunction with diverse classifiers (e.g., SVM, Random Forest, or CNN-based evaluators) would offer a more holistic understanding of its transferability across learning paradigms.

The demonstrated efficiency and stability of DBSMOA suggest strong applicability in real-world environments with limited computational capacity, such as embedded systems, IoT sensor networks, and edge computing applications. Furthermore, its robust convergence behavior and feature selection accuracy make it a promising candidate for biomedical and medical diagnostic domains, where feature dimensionality is high and interpretability is crucial. Future extensions could incorporate adaptive self-tuning mechanisms, enabling DBSMOA to autonomously adjust its control parameters to different data environments, further enhancing its practical utility.

In summary, DBSMOA's dynamic structure, probabilistic binary mapping, and biologically inspired coordination mechanisms collectively establish it as a scalable and reliable optimizer for complex, high-dimensional feature selection tasks.

7 Conclusion and Future Directions

This study presented the *Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA)*, a nature-inspired binary optimization method designed for efficient and scalable feature selection in high-dimensional search spaces. Building upon the original Swordfish Movement Optimization Algorithm (SMOA), DBSMOA integrates three major innovations: (i) dynamic behavioral adaptation through role transitions within the population, (ii) probabilistic binary mapping using a sigmoid-based transfer function, and (iii) an elitist selection mechanism that preserves high-quality solutions across generations. Together, these mechanisms effectively mitigate the common limitations of existing binary metaheuristics—such as premature convergence, fixed control parameters, and sensitivity to local optima—while enhancing adaptability and robustness.

Comprehensive experiments conducted on 52 benchmark datasets confirmed that DBSMOA delivers consistently superior or competitive results compared with twelve state-of-the-art binary optimizers. It achieves high classification accuracy, compact feature subsets, and stable convergence with reduced computational cost. The dynamic balance between exploration and exploitation, guided by population feedback, underpins DBSMOA's scalability and generalization across diverse problem domains.

Future research could explore several realistic extensions of this work. These include the incorporation of adaptive or self-tuning transfer functions to further refine binary decision dynamics, and the hybridization of DBSMOA with lightweight local search strategies to improve convergence precision near optimal solutions. Additionally, extending the current single-objective design into a multi-objective framework may enable simultaneous optimization of competing goals such as accuracy, feature reduction, and computational time. Parameter adaptation—particularly for α and role transition thresholds—also represents a promising avenue for enhancing responsiveness to different data complexities.

All datasets and algorithmic configurations used in this study are publicly available, ensuring full reproducibility of the reported results. All experiments were conducted using fixed random seeds and uniform runtime environments without additional tuning. While DBSMOA demonstrates robust performance, its sensitivity to population size and iteration count suggests potential scalability trade-offs in extremely high-dimensional or streaming data scenarios, which warrant further investigation.

In summary, DBSMOA represents a significant contribution to dynamic binary optimization, combining biologically inspired coordination with adaptive control to achieve efficient, reliable, and interpretable feature selection. Its balanced design and empirical success position it as a strong foundation for future research in adaptive and multi-objective metaheuristic development.

Acknowledgements. This research was supported by the Princess Nourah bint Abdulrahman University Researchers Supporting Project (PNURSP2025R308), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Data Availability Statement

All datasets used in this study are publicly available from established repositories, including the UCI Machine Learning Repository and Kaggle. No proprietary or restricted data were used.

Declarations

Author Contributions

F.R. and S.K. conceived and designed the study. F.R., S.K., and M.E. developed the methodology, while F.R., K.G., and D.K. carried out the experimental investigation. F.R. and S.K. implemented the software. F.R., S.K., and D.K. prepared the original manuscript draft, and all authors contributed to reviewing and editing the final version. Data curation and visualization were performed by F.R., K.G., and A.A. Formal analysis and interpretation of results were conducted by F.R., A.A., and M.E. Project administration was overseen by S.K. All authors read and approved the final manuscript.

Competing Interests

The authors declare that they have no known competing interests, whether financial or non-financial, that could influence the work reported in this paper.

Ethics Approval and Consent to Participate

Not applicable. The study used only publicly available datasets that do not involve human participants or animal subjects.

Consent for Publication

Not applicable. The manuscript does not include any personal or identifiable information.

References

[1] Weiss, S., Indurkhya, N.: Predictive Data Mining: A Practical GuideMorgan. Kaufmann (1998)

[2] Ding, C., Peng, H.: Minimum redundancy feature selection from microarray gene expression data. Journal of bioinformatics and computational biology **3**(02), 185–205 (2005)

[3] Gao, Z., Ding, S.X., Cecati, C.: Real-time fault diagnosis and fault-tolerant control. IEEE Transactions on industrial Electronics **62**(6), 3752–3756 (2015)

[4] Demir, B., Erturk, S.: Hyperspectral image classification using relevance vector machines. IEEE Geoscience and Remote Sensing Letters **4**(4), 586–590 (2007)

[5] Demir, B., Erturk, S.: Hyperspectral image classification using relevance vector machines. IEEE Geoscience and Remote Sensing Letters **4**(4), 586–590 (2007)

[6] Liu, H., Motoda, H.: Feature Selection for Knowledge Discovery and Data Mining vol. 454. Springer, ??? (2012)

[7] Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of machine learning research **3**(Mar), 1157–1182 (2003)

[8] Yu, L., Liu, H.: Feature selection for high-dimensional data: A fast correlation-based filter solution. In: Proceedings of the 20th International Conference on Machine Learning (ICML-03), pp. 856–863 (2003)

[9] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R.P., Tang, J., Liu, H.: Feature selection: A data perspective. ACM computing surveys (CSUR) **50**(6), 1–45 (2017)

[10] Dash, M., Liu, H.: Feature selection for classification. Intelligent data analysis **1**(1-4), 131–156 (1997)

[11] Pudil, P., Novovičová, J., Kittler, J.: Floating search methods in feature selection. Pattern recognition letters **15**(11), 1119–1125 (1994)

[12] Jain, A., Zongker, D.: Feature selection: Evaluation, application, and small sample performance. IEEE transactions on pattern analysis and machine intelligence **19**(2), 153–158 (2002)

[13] Goldberg, D.E.: Optimization, and machine learning. Genetic algorithms in Search (1989)

[14] Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of ICNN'95-international Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995). ieee

[15] Kirkpatrick, S., Gelatt Jr, C.D., Vecchi, M.P.: Optimization by simulated annealing. science **220**(4598), 671–680 (1983)

[16] Yang, X.-S.: Nature-inspired Metaheuristic Algorithms. Luniver press, ??? (2010)

[17] Dorigo, M., Stutzle, T.: Ant colony optimization. mit press, cambridge, ma. MIT Press, Cambridge, MA. (2004)

[18] Glover, F.W., Kochenberger, G.A.: Handbook of Metaheuristics vol. 57. Springer, ??? (2003)

[19] Davis, L.: Handbook of genetic algorithms, van nostrand reinhold (1994)

[20] Mirjalili, S.: How effective is the grey wolf optimizer in training multi-layer perceptrons. Applied intelligence **43**, 150–161 (2015)

[21] Faris, H., Ala'M, A.-Z., Heidari, A.A., Aljarah, I., Mafarja, M., Hassonah, M.A., Fujita, H.: An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks. Information Fusion **48**, 67–83 (2019)

[22] Brest, J., Greiner, S., Boskovic, B., Mernik, M., Zumer, V.: Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. IEEE transactions on evolutionary computation **10**(6), 646–657 (2006)

[23] Hamarashid, H.K., Hassan, B.A., Rashid, T.A.: Modified-improved fitness dependent optimizer for complex and engineering problems. Knowledge-based systems **300**, 112098 (2024)

[24] Poli, R.: Analysis of the publications on the applications of particle swarm optimisation. Journal of Artificial Evolution and Applications **2008**(1), 685175 (2008)

[25] Zaman, E.A.K., Ahmad, A., Mohamed, A.: Adaptive threshold optimisation for online feature selection using dynamic particle swarm optimisation in determining feature relevancy and redundancy. Applied Soft Computing **156**, 111477 (2024)

[26] Author, A.: A sine cosine algorithm guided by elite pool strategy for global optimization. Applied Soft Computing **164**, 111946 (2024) https://doi.org/10.1016/j.asoc.2024.111946

[27] Author, A.: Snow ablation optimizer: A novel metaheuristic technique for numerical optimization and engineering design. Expert Systems with Applications **225**, 120069 (2023) https://doi.org/10.1016/j.eswa.2023.120069

[28] Author, A.: A multi-strategy improved slime mould algorithm for global optimization and engineering design problems. Computer Methods in Applied Mechanics and Engineering **404**, 115764 (2023) https://doi.org/10.1016/j.cma.2023.115764

[29] Author, A.: An enhanced slime mould algorithm based on adaptive grouping technique for global optimization. Expert Systems with Applications **222**, 119877 (2023) https://doi.org/10.1016/j.eswa.2023.119877

[30] Author, A.: Incorporating q-learning and gradient search scheme into jaya algorithm for global optimization. Artificial Intelligence Review **56**(Suppl 3), 3705–3748 (2023) https://doi.org/10.1007/s10462-023-10748-4

[31] Author, A.: A novel hybrid grasshopper optimization algorithm for numerical and engineering optimization problems. Neural Processing Letters **55**(7), 9851–9905 (2023) https://doi.org/10.1007/s11063-023-11439-8

[32] Author, A.: Advancing photovoltaic system design: An enhanced social learning swarm optimizer with guaranteed stability. Computers in Industry **164**, 104209 (2025) https://doi.org/10.1016/j.compind.2024.104209

[33] Madhusudhanan, B., Sumathi, P., Karpagam, N.S., Mahesh, A., Suhi, P.A.P.: An hybrid metaheuristic approach for efficient feature selection. Cluster Computing

22(Suppl 6), 14541–14549 (2019)

[34] Hall, M.A.: Correlation-based feature selection for machine learning. PhD thesis, The University of Waikato (1999)

[35] Kareem, S.S., Mostafa, R.R., Hashim, F.A., El-Bakry, H.M.: An effective feature selection model using hybrid metaheuristic algorithms for iot intrusion detection. Sensors **22**(4), 1396 (2022)

[36] Sugumar, R.: Rough set theory-based feature selection and fga-nn classifier for medical data classification (2019)

[37] Jain, A.K., Chandrasekaran, B.: 39 dimensionality and sample size considerations in pattern recognition practice. Handbook of statistics **2**, 835–855 (1982)

[38] Liaw, A., Wiener, M., *et al.*: Classification and regression by randomforest. R news **2**(3), 18–22 (2002)

[39] Li, W., Chai, Y., Khan, F., Jan, S.R.U., Verma, S., Menon, V.G., Kavita, f., Li, X.: A comprehensive survey on machine learning-based big data analytics for iot-enabled smart healthcare system. Mobile networks and applications **26**, 234–252 (2021)

[40] Lategahn, H., Geiger, A., Kitt, B.: Visual slam for autonomous ground vehicles. In: 2011 IEEE International Conference on Robotics and Automation, pp. 1732–1737 (2011). IEEE

[41] Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)

[42] Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning, pp. 6105–6114 (2019). PMLR

[43] Liu, H., Yu, L.: Toward integrating feature selection algorithms for classification and clustering. IEEE Transactions on knowledge and data engineering **17**(4), 491–502 (2005)

[44] Saeys, Y., Inza, I., Larranaga, P.: A review of feature selection techniques in bioinformatics. bioinformatics **23**(19), 2507–2517 (2007)

[45] Tawhid, A., Teotia, T., Elmiligi, H.: Machine learning for optimizing healthcare resources. In: Machine Learning, Big Data, and IoT for Medical Informatics, pp. 215–239. Elsevier, ??? (2021)

[46] Author, A.: Unlocking new potentials in evolutionary computation with complex network insights: A brief survey. Archives of Computational Methods in Engineering, 1–15 (2025) https://doi.org/10.1007/s11831-025-10254-8

[47] Author, A.: Collective dynamics of particle swarm optimization: A network science perspective. Physica A: Statistical Mechanics and its Applications, 130778 (2025) https://doi.org/10.1016/j.physa.2025.130778

[48] Abd-Alsabour, N., Ramakrishnan, S.: Hybrid metaheuristics for classification problems. Pattern Recognition-Analysis and Applications **10**, 65253 (2016)

[49] Sharma, M., Kaur, P.: A comprehensive analysis of nature-inspired meta-heuristic techniques for feature selection problem. Archives of Computational Methods in Engineering **28**, 1103–1127 (2021)

[50] Du, H., Wang, Z., Zhan, W., Guo, J.: Elitism and distance strategy for selection of evolutionary algorithms. IEEe Access **6**, 44531–44541 (2018)

[51] Mukhopadhyay, A., Maulik, U., Bandyopadhyay, S., Coello, C.A.C.: A survey of multiobjective evolutionary algorithms for data mining: Part i. IEEE Transactions on Evolutionary Computation **18**(1), 4–19 (2013)

[52] Agrawal, P., Abutarboush, H.F., Ganesh, T., Mohamed, A.W.: Metaheuristic algorithms on feature selection: A survey of one decade of research (2009-2019). Ieee Access **9**, 26766–26791 (2021)

[53] Hua, J., Tembe, W.D., Dougherty, E.R.: Performance of feature-selection methods in the classification of high-dimension data. Pattern Recognition **42**(3), 409–424 (2009)

[54] Wei, G., Zhao, J., Feng, Y., He, A., Yu, J.: A novel hybrid feature selection method based on dynamic feature importance. Applied Soft Computing **93**, 106337 (2020)

[55] Salimi, H.: Stochastic fractal search: A powerful metaheuristic algorithm. Knowledge-Based Systems **75**, 1–18 (2015) https://doi.org/10.1016/j.knosys.2014.07.025

[56] Reeves, C.R.: Genetic algorithms. In: Gendreau, M., Potvin, J.-Y. (eds.) Handbook of Metaheuristics, pp. 109–139. Springer, ??? (2010). https://doi.org/10.1007/978-1-4419-1665-5_5

[57] Takieldeen, A., El-kenawy, E.-S., Hadwan, M., Zaki, R.: Dipper throated optimization algorithm for unconstrained function and feature selection. Computers, Materials & Continua **72**(1), 1465–1481 (2022) https://doi.org/10.32604/cmc.2022.026026

[58] Awadallah, M.A., Al-Betar, M.A., Hammouri, A.I., Alomari, O.A.: Binary jaya algorithm with adaptive mutation for feature selection. Arabian Journal for Science and Engineering **45**(12), 10875–10890 (2020) https://doi.org/10.1007/s13369-020-04871-2

[59] Kucukoglu, I.: Binary satin bowerbird optimizer for the set covering problem. In: Calisir, F., Korhan, O. (eds.) Industrial Engineering in the Digital Disruption Era, pp. 73–86. Springer, ??? (2020). https://doi.org/10.1007/978-3-030-42416-9_8

[60] Thaher, T., Heidari, A.A., Mafarja, M., Dong, J.S., Mirjalili, S.: Binary harris hawks optimizer for high-dimensional, low sample size feature selection. In: Mirjalili, S., Faris, H., Aljarah, I. (eds.) Evolutionary Machine Learning Techniques: Algorithms and Applications, pp. 251–272. Springer, ??? (2020). https://doi.org/10.1007/978-981-32-9990-0_12

[61] Azzam, S.M., Emam, O.E., Abolaban, A.S.: An improved differential evolution with sailfish optimizer (desfo) for handling feature selection problem. Scientific Reports **14**(1), 13517 (2024) https://doi.org/10.1038/s41598-024-63328-w

[62] Pereira, J.L.J., Francisco, M.B., Ma, B.J., Gomes, G.F., Lorena, A.C.: Golden lichtenberg algorithm: a fibonacci sequence approach applied to feature selection. Neural Computing and Applications **36**(32), 20493–20511 (2024) https://doi.org/10.1007/s00521-024-10155-9

[63] Xie, W., Wang, L., Yu, K., Shi, T., Li, W.: Improved multi-layer binary firefly algorithm for optimizing feature selection and classification of microarray data. Biomedical Signal Processing and Control **79**, 104080 (2023) https://doi.org/10.1016/j.bspc.2022.104080

[64] Pampara, G., Engelbrecht, A.P., Franken, N.: Binary differential evolution. In: 2006 IEEE International Conference on Evolutionary Computation, pp. 1873–1879 (2006). https://doi.org/10.1109/CEC.2006.1688535

[65] Hussien, A.G., Oliva, D., Houssein, E.H., Juan, A.A., Yu, X.: Binary whale optimization algorithm for dimensionality reduction. Mathematics **8**(10), 1821 (2020) https://doi.org/10.3390/math8101821

[66] Nguyen, B.H., Xue, B., Andreae, P., Zhang, M.: A new binary particle swarm optimization approach: Momentum and dynamic balance between exploration and exploitation. IEEE Transactions on Cybernetics **51**(2), 589–603 (2021) https://doi.org/10.1109/TCYB.2019.2944141

[67] Reddy, K.S., Panwar, L.K., Panigrahi, B., Kumar, R.: A new binary variant of sine–cosine algorithm: Development and application to solve profit-based unit commitment problem. Arabian Journal for Science and Engineering **43**(8), 4041–4056 (2018) https://doi.org/10.1007/s13369-017-2790-x

[68] Kumar, L., Bharti, K.K.: A novel hybrid bpso–sca approach for feature selection. Natural Computing **20**(1), 39–61 (2021) https://doi.org/10.1007/s11047-019-09769-z

[69] Alweshah, M., Khalaileh, S.A., Gupta, B.B., Almomani, A., Hammouri, A.I., Al-Betar, M.A.: The monarch butterfly optimization algorithm for solving feature selection problems. Neural Computing and Applications **34**(14), 11267–11281 (2022) https://doi.org/10.1007/s00521-020-05210-0

[70] El-Mageed, A.A.A., Abohany, A.A., Hosny, K.M.: Enhanced binary kepler optimization algorithm for effective feature selection of supervised learning classification. Journal of Big Data **12**(1), 93 (2025) https://doi.org/10.1186/s40537-025-01125-6

[71] Ranjan, R., Chhabra, J.K.: Automatic feature selection using enhanced dynamic crow search algorithm. International Journal of Information Technology **15**(5), 2777–2782 (2023) https://doi.org/10.1007/s41870-023-01319-2

[72] Khodadadi, N., Khodadadi, E., Al-Tashi, Q., El-Kenawy, E.-S.M., Abualigah, L., Abdulkadir, S.J., Alqushaibi, A., Mirjalili, S.: Baoa: Binary arithmetic optimization algorithm with k-nearest neighbor classifier for feature selection. IEEE Access **11**, 94094–94115 (2023) https://doi.org/10.1109/ACCESS.2023.3310429

[73] Pashaei, E., Pashaei, E.: An efficient binary chimp optimization algorithm for feature selection in biomedical data classification. Neural Computing and Applications **34**(8), 6427–6451 (2022) https://doi.org/10.1007/s00521-021-06775-0

[74] Li, M., Luo, Q., Zhou, Y.: Bgoa-tvg: Binary grasshopper optimization algorithm with time-varying gaussian transfer functions for feature selection. Biomimetics **9**(3) (2024) https://doi.org/10.3390/biomimetics9030187

[75] Shikoun, N.H., Al-Eraqi, A.S., Fathi, I.S.: Bincoa: An efficient binary crayfish optimization algorithm for feature selection. IEEE Access **12**, 28621–28635 (2024) https://doi.org/10.1109/ACCESS.2024.3366495

[76] Liu, G., Guo, Z., Liu, W., Jiang, F., Fu, E.: A feature selection method based on the golden jackal-grey wolf hybrid optimization algorithm. PLOS ONE **19**(1), 1–32 (2024) https://doi.org/10.1371/journal.pone.0295579

[77] Tubishat, M., Alswaitti, M., Mirjalili, S., Al-Garadi, M.A., Alrashdan, M.T., Rana, T.A.: Dynamic butterfly optimization algorithm for feature selection. IEEE Access **8**, 194303–194314 (2020) https://doi.org/10.1109/ACCESS.2020.3033757

[78] Gad, A.G., Sallam, K.M., Chakrabortty, R.K., Ryan, M.J., Abohany, A.A.: An improved binary sparrow search algorithm for feature selection in data classification. Neural Computing and Applications **34**(18), 15705–15752 (2022) https://doi.org/10.1007/s00521-022-07203-7

[79] Akinola, O., Oyelade, O.N., Ezugwu, A.E.: Binary ebola optimization search algorithm for feature selection and classification problems. Applied Sciences **12**(22), 11787 (2022) https://doi.org/10.3390/app122211787

[80] Abualigah, L., Diabat, A.: Chaotic binary group search optimizer for feature selection. Expert Systems with Applications **192**, 116368 (2022) https://doi.org/10.1016/j.eswa.2021.116368

[81] Abd El-Mageed, A.A., Abohany, A.A., Elashry, A.: Effective feature selection strategy for supervised classification based on an improved binary aquila optimization algorithm. Computers & Industrial Engineering **181**, 109300 (2023) https://doi.org/10.1016/j.cie.2023.109300

[82] Thaher, T., Chantar, H., Too, J., Mafarja, M., Turabieh, H., Houssein, E.H.: Boolean particle swarm optimization with various evolutionary population dynamics approaches for feature selection problems. Expert Systems with Applications **195**, 116550 (2022) https://doi.org/10.1016/j.eswa.2022.116550

[83] Ghoneim, S.S.M., Farrag, T.A., Rashed, A.A., El-Kenawy, E.-S.M., Ibrahim, A.: Adaptive dynamic meta-heuristics for feature selection and classification in diagnostic accuracy of transformer faults. IEEE Access **9**, 78324–78340 (2021) https://doi.org/10.1109/ACCESS.2021.3083593

[84] Cinar, A.C.: A novel adaptive memetic binary optimization algorithm for feature selection. Artificial Intelligence Review **56**(11), 13463–13520 (2023) https://doi.org/10.1007/s10462-023-10482-8

[85] El-Sayed, M., Ali, A., Sami, D., Amal, H., Sarah, A., Abdelaziz, A., Abdelhameed, I., Marwa, M.: A novel binary swordfish movement optimization algorithm (bsmoa) for efficient feature selection. Fusion: Practice and Applications, 170–186 (2025) https://doi.org/10.54216/FPA.190213

[86] Takieldeen, A.E., El-kenawy, E.-S.M., Hadwan, M., Zaki, R.M.: Dipper throated optimization algorithm for unconstrained function and feature selection. Comput. Mater. Contin **72**, 1465–1481 (2022)

[87] Mirjalili, S., Mirjalili, S.M., Lewis, A.: Grey wolf optimizer. Advances in engineering software **69**, 46–61 (2014)

[88] Heidari, A.A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., Chen, H.: Harris hawks optimization: Algorithm and applications. Future generation computer systems **97**, 849–872 (2019)

[89] Mirjalili, S.: Sca: a sine cosine algorithm for solving optimization problems. Knowledge-based systems **96**, 120–133 (2016)

[90] Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of ICNN'95-international Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995). ieee

[91] Mirjalili, S., Lewis, A.: The whale optimization algorithm. Advances in engineering software **95**, 51–67 (2016)

[92] Salimi, H.: Stochastic fractal search: a powerful metaheuristic algorithm. Knowledge-based systems **75**, 1–18 (2015)

[93] Feoktistov, V.: Differential Evolution. Springer, ??? (2006)

[94] Moosavi, S.H.S., Bardsiri, V.K.: Satin bowerbird optimizer: A new optimization algorithm to optimize anfis for software development effort estimation. Engineering Applications of Artificial Intelligence **60**, 1–15 (2017)

[95] Rao, R.V.: Jaya: an advanced optimization algorithm and its engineering applications (2019)

[96] Johari, N.F., Zain, A.M., Noorfa, M.H., Udin, A.: Firefly algorithm for optimization problem. Applied Mechanics and Materials **421**, 512–517 (2013)

[97] Sivanandam, S., Deepa, S., Sivanandam, S., Deepa, S.: Genetic Algorithms. Springer, ??? (2008)

[98] Jain, A.K., Duin, R.P.W., Mao, J.: Statistical pattern recognition: A review. IEEE Transactions on pattern analysis and machine intelligence **22**(1), 4–37 (2000)

[99] Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine learning **46**, 389–422 (2002)