Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Dynamic Swordfish Movement Optimization Algorithm for Feature Selection

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

[1]Department of Communications and Electronics, Delta Higher Institute of Engineering and Technology, Mansoura 35111, Egypt (e-mail: faris.hamdi.rizk@gmail.com
[2]Computer Science and Intelligent Systems Research Center, Blacksburg 24060, Virginia, USA (e-mail: khsherif@jcsis.org)
[3]Faculty of Artificial Intelligence, Delta University for Science and Technology, Mansoura, Egypt (e-mail: mmm@ieee.org)
[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 (e-mails: dskhafga@pnu.edu.sa, aaalhussan@pnu.edu.sa)
[6]Department of Programming, School of Information and Communications Technology (ICT), Bahrain Polytechnic, PO Box 33349, Isa Town, Bahrain(e-mail: skenawy@ieee.org)
[7]Applied Science Research Center. Applied Science Private University, Amman, Jordan

Corresponding author(s): El-Sayed M. El-kenawy (e-mail: skenawy@ieee.org), Marwa M. Eid (e-mail: mmm@ieee.org).

Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R308), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia

ABSTRACT Feature selection represents a crucial preprocessing step in machine learning pipelines, particularly when dealing with high-dimensional datasets that often contain redundant or irrelevant information. To address the challenge of efficiently selecting informative features while optimizing classification accuracy, this paper introduces the Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA), a novel binary metaheuristic inspired by the foraging behavior of swordfish. DBSMOA enhances the original Swordfish Movement Optimization Algorithm (SMOA) by incorporating dynamic behavioral mechanisms that adaptively balance exploration and exploitation throughout the search process. The proposed algorithm employs a binary encoding strategy based on sigmoid probabilistic mapping and dynamically alternates agent roles using real-time performance metrics and elitist selection strategies. To assess its efficacy, DBSMOA is extensively evaluated on diverse benchmark datasets and compared against 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 consistently achieves superior performance in classification accuracy, feature reduction, and computational efficiency. These findings highlight the robustness and adaptability of DBSMOA for binary optimization problems, as well as its practical potential for high-dimensional data analytics.

INDEX TERMS Dynamic Binary Optimization, Feature Selection, Metaheuristic Algorithms, Swordfish Movement Optimization, High-Dimensional Data Analysis

I. INTRODUCTION

IN recent years, the proliferation of high-dimensional datasets across domains such as medical informatics [1], computational biology [2], fault diagnostics [3], remote sensing [4], and cyber-physical systems [5] has necessitated the development of efficient computational tools capable of extracting meaningful patterns from complex and voluminous data. In these domains, datasets often contain hundreds or thousands of features, not all of which contribute equally to the underlying decision-making processes. Redundant, irrelevant, or noisy features not only increase computational costs [6], but may also obscure the performance of machine learning classifiers [7], leading to overfitting and reduced generalization [8]. Feature selection, which aims to identify the most informative subset of features while discarding the rest, has become an indispensable step in modern data analysis pipelines [9], [10].

The task of feature selection is inherently a combinato-

rial optimization problem [11], wherein the objective is to explore the power set of all available features to determine the subset that yields optimal predictive performance. Given the exponential growth of the search space concerning the number of features, exhaustive search techniques become computationally infeasible even for moderately sized datasets [12]. Consequently, a vast body of research has turned toward metaheuristic algorithms—optimization techniques inspired by natural processes such as evolution [13], swarm behavior [14], and thermodynamics [15]—for navigating these expansive and rugged search landscapes. Metaheuristics offer a flexible and problem-independent approach to efficiently approximate near-optimal solutions [16], making them particularly suitable for real-world problems where exact methods are computationally prohibitive [17].

Among these, binary metaheuristic algorithms have received considerable attention due to the discrete nature of feature selection itself [18]. In binary optimization frameworks, each candidate solution is typically encoded as a binary vector, where each bit represents the inclusion (1) or exclusion (0) of a corresponding feature. Several continuous metaheuristics have been adapted to operate in the binary domain, leading to the development of Binary Particle Swarm Optimization (bPSO), Binary Genetic Algorithms (bGA) [19], Binary Grey Wolf Optimizer (bGWO) [20], and similar approaches. These algorithms offer powerful capabilities for traversing complex search spaces, but they often suffer from a lack of robustness and reliability when applied to real-world, high-dimensional datasets. Common issues include convergence to local optima [21], unstable selection patterns across runs, and a loss of population diversity during the search process.

A critical limitation observed across many binary metaheuristics lies in their static nature. Most existing algorithms adopt fixed control parameters, and predefined movement rules that remain unchanged throughout the optimization process. These static behaviors can lead to premature convergence, lack of diversity in the population, and sensitivity to initial conditions. As the search progresses, the algorithm's inability to adjust its behavior in response to stagnation or suboptimal convergence trends hampers its effectiveness in locating globally optimal solutions. This challenge is further exacerbated in binary domains, where the transition dynamics—i.e., how a solution moves from one binary state to another—are inherently constrained and prone to loss of exploratory richness [22].

To address these challenges, researchers have increasingly explored the incorporation of adaptivity and feedback mechanisms into optimization strategies. This has given rise to the paradigm of *dynamic metaheuristic optimization*, which emphasizes real-time algorithmic adaptability based on search progress and population characteristics. In dynamic metaheuristics, components such as mutation rates, movement equations, or role assignments are not fixed. Still, they are instead modulated in response to metrics, including fitness variance [23], diversity scores [24], or convergence rate. Such flexibility enables the algorithm to fine-tune its balance between exploration and exploitation in real time, thereby enhancing search effectiveness and solution quality. Furthermore, adaptivity helps improve the generalizability and robustness of selected feature subsets, especially when applied to noisy or heterogeneous datasets [25].

In the context of binary optimization for feature selection, the integration of dynamic mechanisms enables the algorithm to modulate its behavior in a context-aware fashion [26]. For example, agents within a swarm may adapt their movement rules based on local fitness landscapes, population diversity, or convergence stagnation metrics [27]. Such dynamic behaviors can introduce feedback-driven learning, enhance robustness, and mitigate the risks of stagnation and redundancy, particularly in high-dimensional and noisy search environments [28]. Furthermore, adaptive algorithms often exhibit improved generalization performance by avoiding overfitting to spurious feature interactions that static methods may exploit [29]. This property is particularly relevant in domains such as biomedical data analysis, where small sample sizes and high feature counts make the risk of overfitting particularly acute [30].

A further consideration in the design of modern optimization algorithms is scalability and computational efficiency [31]. With increasing data dimensionality and the real-time constraints of applications such as IoT-enabled health monitoring [32], autonomous vehicles [33], and large-scale text and image analysis [34], algorithms must not only be accurate but also lightweight and computationally practical [35]. In such scenarios, traditional exhaustive or greedy search methods quickly become impractical [36], while dynamic, feedback-driven metaheuristics offer a compelling alternative [37]. Scalability is also closely tied to resource consumption—such as CPU usage and memory footprint—which can be a critical bottleneck in deployment environments with constrained computing capabilities [38].

In response to these considerations, the present study proposes a new dynamic binary metaheuristic for feature selection. The foundation of this method lies in the Swordfish Movement Optimization Algorithm (SMOA), a swarm-based optimizer inspired by the foraging behavior and coordinated attacks of swordfish in natural marine environments. While the original SMOA was designed for continuous optimization tasks, this work reformulates its movement strategies, social interactions, and fitness evaluation mechanisms within a discrete binary framework. More importantly, the proposed algorithm, termed the *Dynamic Binary Swordfish Movement Optimization Algorithm* (DBSMOA), incorporates adaptive dynamics at multiple levels of the algorithm's structure.

Unlike conventional static binary optimizers, DBSMOA divides the swarm into distinct functional groups that alternate between exploration and exploitation roles [39]. These roles are not fixed but evolve during optimization based on empirical criteria such as agent performance, population fitness distribution, and iteration progress [40]. Additionally, the algorithm implements elitist selection strategies to pre-

serve historically optimal solutions and prevent their degradation through random fluctuations [41]. These dynamic features collectively aim to enhance convergence stability, maintain population diversity, and achieve high-quality solutions across multiple independent runs [42].

The design of DBSMOA is motivated by the need to bridge the gap between solution accuracy and convergence reliability in binary feature selection [43]. By embedding dynamism into the search behavior, the algorithm can better adapt to the idiosyncrasies of each dataset, particularly in scenarios characterized by large numbers of features, sparse data representations, or overlapping class boundaries [44]. As such, DBSMOA represents a step forward in the development of responsive and resilient binary optimizers that go beyond the limitations of static behavior. It offers a unified, biologically inspired, and performance-aware approach to feature selection, capable of balancing accuracy, efficiency, and generalizability in high-dimensional search spaces [45].

The key contributions of this paper are summarized as follows:

- A new metaheuristic algorithm, termed Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA), is proposed by extending the original Swordfish Movement Optimization Algorithm (SMOA) with dynamic behavior and binary search capabilities.
- The algorithm incorporates adaptive exploration and exploitation mechanisms by dynamically partitioning the population into functional subgroups based on feedback from the fitness landscape and progress in iteration.
- A binary conversion strategy is integrated into the continuous SMOA framework using a sigmoid-based probabilistic mapping, enabling effective application to binary optimization tasks such as feature selection.
- An elitism mechanism is employed to preserve the best-performing solutions across generations, enhancing convergence reliability and avoiding solution degradation.
- The proposed approach is evaluated on benchmark datasets for feature selection and compared against several established binary metaheuristics to assess its effectiveness in high-dimensional binary optimization scenarios.

The remainder of this paper is structured as follows: Section II reviews related work on feature selection and binary metaheuristics. Section III presents the proposed algorithm in detail, including the original SMOA, its dynamic enhancement, and the final binary dynamic variant. Section IV describes the experimental setup and results obtained from benchmark datasets. Section V provides a comprehensive discussion and analysis of the findings. Finally, Section VI concludes the paper and outlines potential directions for future research.

II. LITERATURE REVIEW

This section surveys previous research related to our paper, organized into key categories of feature selection techniques: Embedded, Wrapper, and Filter methods. Each is discussed in terms of its conceptual foundation, efficiency, and applications.

Feature Selection (FS) aims to reduce data dimensionality by selecting the most relevant features from a dataset. This process enhances model performance, improves computational efficiency, and prevents overfitting. FS methods are generally classified into three main categories: Embedded Methods, Wrapper Methods, and Filter Methods.

In broader terms, feature selection techniques can be categorized based on the availability of labels and the nature of the learning process: supervised, unsupervised, and semi-supervised. As illustrated in Figure 1, supervised methods are further subdivided into filter, wrapper, and embedded approaches. This hierarchical classification helps contextualize the landscape of existing methods and informs the organization of this literature review.

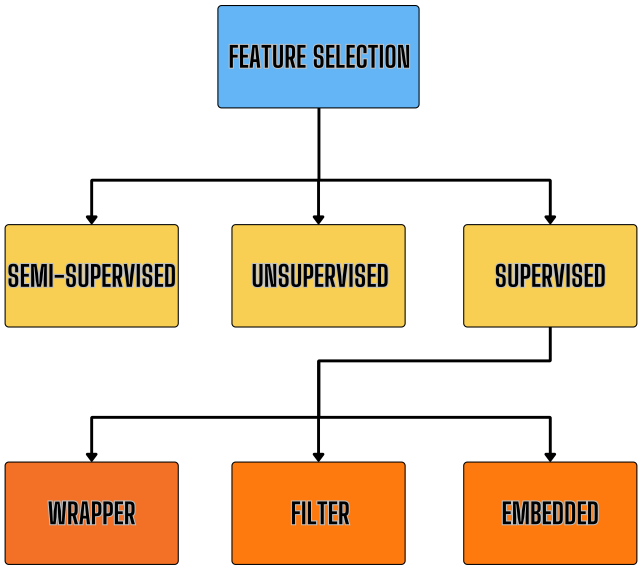

FIGURE 1: Taxonomy of feature selection methods based on learning supervision and evaluation strategy.

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

1) Embedded Methods

Embedded feature selection methods integrate the selection process directly into the model training phase, allowing the algorithm to simultaneously learn which features are most relevant while building the predictive model. This tight integration promotes computational efficiency and can yield highly task-specific feature subsets, especially in domains involving high-dimensional datasets. One of the most prominent embedded methods is the Binary JAYA algorithm (BJAM) [49], which adapts the continuous JAYA optimizer for binary feature spaces using a sinusoidal transfer function and an adaptive mutation strategy. BJAM demonstrated significant improvements in classification accuracy and fea-

TABLE 1: Condensed Summary of Feature Selection Methods in Recent Studies

Method Type	Study Summary	Methodological Highlights
Embedded	[46]: Fractal-based EA avoids local minima in optimization tasks.	Uses fractal diffusion for better exploration.
	[47]: GA implementation guidance for FS.	Foundational embedded FS concepts.
	[48]: DTO inspired by bird motion.	Fast, efficient search strategy.
	[49]: BJAM (Binary JAYA) with adaptive mutation.	Boosts FS accuracy via binary tuning.
	[50]: BSBO for Set Covering Problem.	Binary conversion + repair operator.
	[51]: Binary HHO for high-dim FS.	Dynamic, stable wrapper integration.
	[52]: DESFO combines DE and Sailfish.	Achieves 100% accuracy on benchmarks.
	[53]: GLA uses Fibonacci-guided scaling.	Compact features, high accuracy.
Wrapper	[54]: mRMR + firefly for gene FS.	Two-stage hybrid filtering.
	[55]: DE with angle modulation.	Translates continuous FS to binary.
	[56]: bWOA with sigmoid/tanh functions.	FS across 24 datasets.
	[57]: Dynamic sticky BPSO.	Improved search via binary movement.
	[58]: BSCA for power scheduling.	Uses sine/cosine for FS modeling.
	[59]: HBPSOSCA merges BPSO + SCA.	Enhances clustering-based FS.
	[60]: MBO + k-NN; 93% accuracy avg.	Strong global-local balance.
	[61]: BKOA-MUT; FS for 25 UCI sets.	Combines Kepler law + mutation.
	[62]: EDCSA with k-means for FS.	Improved clustering performance.
	[63]: BAOA with sigmoid mapping.	Outperforms BPSO, BGA, BDF.
	[64]: Binary ChOA for gene FS.	Competes with top wrapper methods.
	[65]: BGOA-TVG with Gaussian transfer.	Strong global search; fast convergence.
	[66]: BinCOA with refracted learning and crisscross strategy.	Outperforms 7 wrappers; Wilcoxon test applied.
	[67]: GJO-GWO hybrid FS algorithm.	Multi-strategy fusion with Lagrange interpolation.
	[68]: DBOA using mutation-based local search.	Beats other methods on 20 UCI datasets.
Filter	[69]: iBSSA with local search.	Compact subsets, high accuracy.
	[70]: BEOSA across 22 datasets.	Minimal features, top results.
	[71]: CGSO with chaotic maps.	FS + classifier model simplification.
	[72]: IBAO with local refinement.	Achieves 100% accuracy.
	[73]: BPSO-TEPD with EPD.	Better convergence on FS.
	[74]: AD-PRS-Guided WOA for faults.	97% accuracy in diagnosis.
	[75]: AMBO with memetic logic.	Outperforms 10+ FS algorithms.

ture reduction across 22 real-world benchmark datasets. Similarly, the Binary Harris Hawks Optimizer (HHO) [51] was adapted to handle binary search spaces by incorporating exploration mechanisms such as greedy selection and Levy flight. Its performance on high-dimensional biomedical datasets confirmed its effectiveness in producing stable and compact feature subsets.

Several hybrid and nature-inspired embedded methods have also shown promising results. The Differential Evolution and Sailfish Optimizer hybrid (DESFO) [52] achieved 100% classification accuracy in specific benchmarks by incorporating feature selection into a hybrid optimization loop, utilizing classifiers such as k-NN and Random Forest. The Golden Lichtenberg Algorithm (GLA) [53] employs Fibonacci-based scaling to enhance search behavior, providing compact feature sets and high predictive accuracy on UCI datasets. Other contributions include the Fractal-based Evolutionary Algorithm (FEA) [46], which utilizes random diffusion-inspired search to enhance exploration, and the Dipper Throated Optimization (DTO) algorithm [48], which employs bird-inspired motion modeling in metaheuristic search. Although not initially focused on FS, these algorithms offer adaptable frameworks for binary search problems. Additionally, the Satin Bowerbird Optimizer (BSBO) [50] was successfully applied to discrete binary optimization tasks, such as the Set Covering Problem, demonstrating the potential for FS scenarios. Foundational work such as the Genetic Algorithm framework outlined in [47] continues to influence embedded FS models by providing theoretical and implementation insights into encoding, selection, and crossover strategies.

2) Wrapper Method

Wrapper feature selection methods rely on the predictive performance of a learning algorithm to evaluate different feature subsets, making them highly accurate but often computationally intensive. These methods typically explore the feature space more thoroughly and are closely tied to classifiers during evaluation. Numerous studies have introduced wrapper-based strategies using metaheuristic algorithms to optimize feature selection. The Chimp Optimization Algorithm (ChOA) [64], adapted for binary domains, incorporates both S-shaped and V-shaped transfer functions, as well as crossover operators, to enhance exploration. Validated on biomedical, text, and image datasets, it outperformed several established algorithms, such as GA, PSO, and FA, in both classification accuracy and dimensionality reduction. Similarly, the Enhanced Binary Kepler Optimization Algorithm (BKOA-MUT) [61] combines Keplerian motion dynamics with differential evolution mutation operators, yielding high accuracy and compact feature sets across 25 UCI benchmark datasets using classifiers like k-NN and SVM. The Improved Binary Aquila Optimization (IBAO) algorithm [72] also merges global exploration with local refinement to achieve up to 100% classification accuracy in some benchmarks while reducing feature sets by up to 92%. These studies

demonstrate how hybrid and enhanced wrapper approaches can effectively balance exploration-exploitation trade-offs.

Additional wrapper-based FS methods have introduced novel binary adaptation mechanisms. For instance, the Binary Arithmetic Optimization Algorithm (BAOA) [63] utilizes a sigmoid transfer function to enable binary encoding and is evaluated using a k-NN classifier across 18 UCI datasets, outperforming methods such as BPSO, BGA, and BDF. The Monarch Butterfly Optimization (MBO) algorithm [60], when combined with k-NN, achieved an average accuracy of 93% across 18 datasets, demonstrating its ability to reduce the number of selected features without compromising predictive power. In the clustering domain, the Enhanced Dynamic Cluster Search Algorithm (EDCSA) [62] and the Hybrid BPSO–Sine Cosine Algorithm (HBPSOSCA) [59] demonstrate the utility of wrapper feature selection (FS) in improving unsupervised tasks. Angle modulation has been effectively employed in adapting Differential Evolution to binary search spaces [55], thereby reducing the complexity of traditional binary optimization. Furthermore, wrapper-based improvements to Whale Optimization [56] and Sticky Binary PSO [57] have enabled better control over feature subset size and classifier performance. The Binary Sine–Cosine Algorithm (BSCA) [58], although developed for unit commitment, demonstrates strong potential in FS tasks due to its binary transformation mechanism and dynamic search behavior. Finally, a hybrid two-phase approach combining mRMR filtering with multilayer binary firefly optimization [54] has shown impressive classification accuracy and dimensionality reduction, especially in high-dimensional biomedical data.

Recent developments in wrapper-based methods continue to demonstrate the importance of hybridization and dynamic strategy design for efficient feature selection. For instance, the Binary Grasshopper Optimization Algorithm with Time-Varying Gaussian transfer functions (BGOA-TVG) [65] introduces a dynamic transfer mechanism to map continuous search spaces into binary domains. This approach enhances the balance between exploration and exploitation, demonstrating high classification accuracy across the UCI, DEAP, and EPILEPSY datasets. It outperforms traditional S-shaped and V-shaped binary transfer functions and several recent swarm intelligence algorithms.

Similarly, the Binary Crayfish Optimization Algorithm (BinCOA) [66] presents a novel adaptation of the Crayfish Optimization Algorithm (COA) tailored for binary search. The integration of refracted opposition-based learning during initialization and a crisscross strategy during exploitation enhances both convergence and solution diversity. BinCOA was validated on 30 benchmark datasets and outperformed seven contemporary FS methods in terms of accuracy, subset compactness, and statistical significance.

Another hybrid approach is the GJO-GWO algorithm [67], which fuses Golden Jackal Optimization (GJO) with the Grey Wolf Optimizer (GWO) using Lagrange interpolation. It consistently achieves lower fitness values and higher classification accuracy compared to traditional methods in solving both benchmark optimization functions and FS tasks. Its multi-strategy framework reflects a growing trend toward composite search models for improved FS outcomes.

The Dynamic Butterfly Optimization Algorithm (DBOA) [68] is proposed as an enhancement over the original Butterfly Optimization Algorithm (BOA). It introduces a Local Search Algorithm based on Mutation (LSAM) to address BOA's weakness in stagnating and maintaining diversity in high-dimensional FS problems. The DBOA has been tested on 20 UCI datasets and consistently shows superior classification performance, reduced subset size, and better convergence compared to its counterparts.

3) Filter Method

Filter-based feature selection methods evaluate the relevance of features independently of any specific machine learning model. They use statistical measures to rank features and eliminate those deemed irrelevant or redundant before the classification stage. These methods are highly valued for their simplicity and scalability, particularly when working with extremely high-dimensional data. Recent advancements have extended filter techniques through the use of metaheuristic algorithms adapted for binary optimization. For instance, the Improved Binary Sparrow Search Algorithm (iBSSA) [69] utilizes local search and random repositioning to strike a balance between exploration and exploitation in binary search spaces. Evaluated using classifiers such as k-NN, SVM, and Random Forest, iBSSA consistently produced compact feature subsets while maintaining high classification accuracy across 18 benchmark datasets. Similarly, the Binary Ebola Optimization Search Algorithm (BEOSA) [70] applied newly designed S-shaped and V-shaped transfer functions to model binary transitions. Tested on 22 benchmark datasets, BEOSA achieved the best feature subset results on multiple datasets while minimizing feature count.

Other filter approaches have focused on enhancing search behavior or complexity control. The Chaotic Group Search Optimizer (CGSO) [71] integrates five chaotic maps to guide binary exploration for feature selection (FS), successfully improving classification accuracy and reducing model complexity across 20 UCI datasets. The Improved Binary Aquila Optimization (IBAO) algorithm [72] enhances both global and local search, achieving up to 100% classification accuracy with up to 92% feature reduction across 18 datasets. Furthermore, the Evolutionary Population Dynamics-based BPSO-TEPD [73] incorporates natural selection mechanisms to enhance the convergence and stability of wrapper-filter hybrids. While designed for general binary optimization, the Adaptive Dynamic Polar Rose-Guided Whale Optimization Algorithm (AD-PRS-Guided WOA) [74] demonstrated high classification accuracy (97.1%) in transformer fault diagnosis by tuning classification parameters with FS techniques. Lastly, the Adaptive Memetic Binary Optimization (AMBO) method [75] utilized logic-gate-based local search and adaptive mutation strategies. It significantly outperformed sev-

eral well-known feature selection (FS) methods across 21 datasets, particularly in reducing classification error rates and optimizing feature subset size. Collectively, these methods highlight the evolution of traditional filter techniques into more adaptive and hybridized systems that combine speed with increased search intelligence.

To provide a comprehensive overview of foundational and recent advancements in feature selection research, Table 2 summarizes all 26 referenced studies. Each entry outlines the study's focus, methodological contributions, key limitations, and datasets used. This table serves as a comparative reference to highlight performance trends, common challenges, and the breadth of applications across biomedical, energy, clustering, and general classification tasks.

Research Gap

Despite the progress in binary metaheuristic algorithms for feature selection, several limitations persist in the current literature. Most existing algorithms operate under static control mechanisms, where exploration and exploitation dynamics are fixed throughout the optimization process. This often leads to premature convergence, reduced diversity, and suboptimal feature subsets—especially in high-dimensional, complex datasets. While some studies have introduced adaptive or hybrid enhancements, they frequently lack a unified strategy that incorporates dynamic population behavior, elitism, and robust binary conversion mechanisms in a single framework.

Moreover, many prior methods apply either basic binary transformations or fixed transfer functions without considering the influence of adaptive behavioral roles or context-aware transitions between search phrases. Few algorithms explicitly model role-switching or adjust swarm dynamics in response to real-time population performance. This leaves a significant opportunity for improvement in achieving both convergence quality and solution stability.

To bridge these gaps, our work introduces the Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA), which integrates dynamic role transitions, elitist memory retention, and a sigmoid-based probabilistic mapping tailored for binary feature selection. Unlike prior methods, DBSMOA adaptively balances exploration and exploitation based on population feedback while maintaining diversity and convergence stability. This unified and biologically inspired design fills a critical void in existing binary optimization strategies for scalable and reliable feature selection.

III. THE PROPOSED DYNAMIC SWORDFISH MOVEMENT OPTIMIZATION ALGORITHM (DSMOA)

A. THE ORIGINAL SWORDFISH MOVEMENT OPTIMIZATION ALGORITHM (SMOA)

The Swordfish Movement Optimization Algorithm (SMOA) is a recently developed nature-inspired metaheuristic that draws its behavioral model from the synchronized, agile, and strategic hunting dynamics of swordfish in marine ecosystems [76]. These predators exhibit a high level of coordination, relying on both individual agility and group intelligence to target and isolate schools of prey. This dual emphasis on broad-area exploration and sharply focused exploitation makes swordfish an ideal biological metaphor for global optimization techniques, especially those navigating multimodal, high-dimensional search landscapes.

The core premise of SMOA lies in its ability to simulate natural foraging dynamics through three principal behavioral mechanisms: *exploration*, *exploitation*, and *elimination*. Each phase governs a unique mode of interaction with the solution space, encoded mathematically using dynamically adaptive control parameters. The agents (swordfish) iteratively update their positions within the search space based on collective behavior, local fitness evaluations, and adaptive rules that mimic real-world interactions.

1) Exploration Phase

The exploration phase is designed to facilitate global search capabilities by allowing agents to traverse wide regions of the solution space. The positional update rule for this phase incorporates a mixture of sinusoidal motion, random direction changes, and adaptive scaling:

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

where $\vec{T}(t)$ denotes the position vector of an agent at iteration t, a is a control constant in the range $[0, 1]$, and Θ is an angular parameter used to introduce nonlinear movement components. The dynamic coefficients K and Z are computed as:

$$K = \left(1 + \frac{(\text{iteration})^2}{\text{iteration}}\right)^2, \quad Z = \left(X + \frac{2 \cdot (\text{iteration})^2}{N}\right)^2, \tag{2}$$

where X is a random number drawn from $[1, 2]$, and N is the maximum number of iterations.

2) Velocity and Collective Movement Updates

To refine the direction and speed of exploration, auxiliary vectors $\vec{M}(t+1)$ and $\vec{J}(t+1)$ are introduced:

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

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

where r is a dynamic randomization parameter calculated as:

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

with h being a constant scalar that governs motion intensity.

TABLE 2: Overview of Feature Selection Techniques in Recent Studies

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

3) Exploitation Phase

When agents identify promising regions of the solution space, the exploitation phase focuses on refining these regions. The updated position vector is determined by:

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

, thereby reinforcing convergence toward high-fitness regions.

4) Elimination Phase

To preserve population diversity and prevent stagnation, the elimination phase introduces random disruptions based on global parameters:

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

where g is a constant scaling coefficient, ensuring controlled diversification during stagnation.

SMOA Pseudocode

The step-by-step flow of SMOA can be summarized as follows:

Algorithm 1 Swordfish Movement Optimization Algorithm (SMOA)

```
Initialize population of swordfish S_i (i = 1, 2, ..., n) randomly in the search space
Initialize parameters: N, a, h, Θ, max iterations
for each iteration t = 1 to N do
    for each agent i do
        Evaluate fitness f(S_i)
        Compute dynamic coefficients K, Z, and r
        Update T_i(t + 1) using Equation (1)
        Update M_i(t + 1) and J_i(t + 1) using Equations (3) and (4)
        Compute new position S_i(t + 1):
        if in exploration phase then
            Use exploration equation
        else if in exploitation phase then
            Use exploitation equation
        else if stagnation detected then
            Use elimination equation
        end if
    end for
    Update global best solution S_best
end for
return S_best
```

This modular and behaviorally rich formulation enables SMOA to dynamically switch between exploration, exploitation, and diversification modes, allowing for robust search and reliable convergence even in highly complex optimization problems.

B. DYNAMIC SWORDFISH MOVEMENT OPTIMIZATION ALGORITHM (DSMOA)

The Dynamic Swordfish Movement Optimization Algorithm (DSMOA) represents an enhanced extension of the classical Swordfish Movement Optimization Algorithm (SMOA), specifically tailored to address the limitations observed in static metaheuristic optimizers. While the original SMOA simulates the strategic foraging behavior of swordfish using continuous movement dynamics, its static control structures and homogeneous population behavior may limit adaptability in evolving or complex search environments. DSMOA introduces adaptive behavioral dynamics, population stratification, and elitist selection to overcome these constraints, offering a responsive and resilient search mechanism for high-dimensional optimization problems.

1) Motivation for Dynamic Behavior

Traditional metaheuristics often rely on static exploration and exploitation mechanisms, where the balance between global search and local refinement remains fixed throughout the optimization process. This static paradigm can result in premature convergence, excessive exploitation of suboptimal regions, and an inability to respond effectively to the dynamic characteristics of the fitness landscape. In high-dimensional or non-stationary optimization problems, these weaknesses result in variability in solution quality and reduced generalization performance.

DSMOA is designed to mitigate these deficiencies by embedding dynamic behavior directly into the search process. By allowing parameters, roles, and strategies to evolve in response to population performance and iteration progress, the algorithm remains adaptive and efficient across different stages of optimization.

2) Exploration vs. Exploitation Adaptation

DSMOA regulates the transition between exploration and exploitation not via static probability thresholds but through performance-driven criteria. At each iteration, the algorithm assesses population diversity, convergence stagnation, and variance in fitness. If diversity is low or the population exhibits early saturation, a shift toward exploration is initiated. Conversely, if convergence indicators suggest that a focused search is appropriate, exploitation mechanisms are prioritized.

This adaptive balancing mechanism enhances robustness by adjusting search intensity based on real-time feedback rather than relying on predefined iteration milestones or deterministic switching.

3) Dynamic Group Formation and Role Transitions

Inspired by coordinated marine hunting strategies, DSMOA dynamically partitions the population into two primary groups: *explorers* and *exploiters*. These groups are not fixed. Instead, agents are reassigned at each iteration based on performance metrics such as relative fitness and historical improvement rate.

Explorers are tasked with traversing broader regions of the solution space using diversified motion and stochastic updates. Exploiters concentrate search efforts near elite agents using localized refinement strategies derived from SMOA's exploitation mechanics. This dynamic reallocation of roles maintains a healthy balance between exploration and

exploitation throughout the optimization process, promoting adaptive specialization within the population.

4) Elitism and Convergence Refinement

To preserve progress and avoid the degradation of high-quality solutions, DSMOA incorporates an elitism mechanism. The best-performing individuals are recorded in an elite archive and reintroduced periodically into the population or used to guide exploitation strategies. This practice ensures that strong solutions are not lost due to random fluctuations or role transitions.

Additionally, convergence refinement is achieved through localized updates around elite solutions. These refinement steps are based on adaptive step sizes, directional updates influenced by elite trajectories, and neighborhood-based exploitation. These strategies fine-tune already strong candidates, improving convergence reliability and reducing fitness variance across runs.

5) Updated Movement Strategies

The core equations from SMOA are retained in DSMOA but modified with dynamic coefficients and behavioral transitions. Instead of applying uniform update equations to all agents, DSMOA differentiates update rules by group. Explorers utilize adaptive versions of the sinusoidal and randomized exploration equations, whereas exploiters employ focused movement strategies that incorporate elite-influenced motion and precision controls.

This design ensures that each agent's behavior is context-aware and modulated in real time based on its assigned role and the algorithm's global state.

DSMOA Algorithmic Framework

The overall flow of the algorithm is outlined below:

This algorithmic framework ensures that DSMOA not only adapts to the problem landscape but also evolves with it, offering a principled and biologically inspired mechanism for solving complex continuous-domain optimization problems in dynamic environments.

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, etc.
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 elite archive
    Compute diversity and fitness variance of population
    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 exploitation update guided by elite solutions
        Perform convergence refinement with local adjustments
    end for
    Update global best if improved
end for
return $\vec{S}_{\text{best}}$

C. DYNAMIC BINARY SWORDFISH MOVEMENT OPTIMIZATION ALGORITHM (DBSMOA)

The Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA) is a discrete variant of the proposed Dynamic Swordfish Movement Optimization Algorithm (DSMOA), re-engineered specifically for binary search spaces. While DSMOA is designed for continuous domains, DBSMOA adapts its dynamic behavior for combinatorial optimization tasks, such as feature selection, where solutions are represented as binary vectors indicating the inclusion or exclusion of features. DBSMOA inherits the core behavioral phases of DSMOA—exploration, exploitation, and elimination—but redefines them using binary-compatible mathematical mechanisms. The core innovation lies in the binary encoding of position vectors and the transformation of movement equations through probabilistic mapping, enabling the algorithm to retain biologically inspired swarm behavior within a discrete context.

Figure 2 illustrates the overall flow of DBSMOA, including initialization, dynamic coefficient computation, movement updates, binary transformation using a sigmoid function, and elitist solution updating. This process continues iteratively until convergence or the maximum number of iterations is reached.

1) Binary Encoding of Positions

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

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

where d is the total number of features. A bit $s_{ij} = 1$ indicates that the j^{th} feature is selected in the i^{th} solution, while $s_{ij} = 0$ denotes exclusion. The algorithm searches for binary strings that yield feature subsets, optimizing classification accuracy while minimizing dimensionality.

2) Mathematical Modeling of DBSMOA

To translate the continuous movement dynamics of DSMOA into binary space, a two-step transformation is applied. First, standard DSMOA equations generate an intermediate continuous movement vector $\vec{v}$:

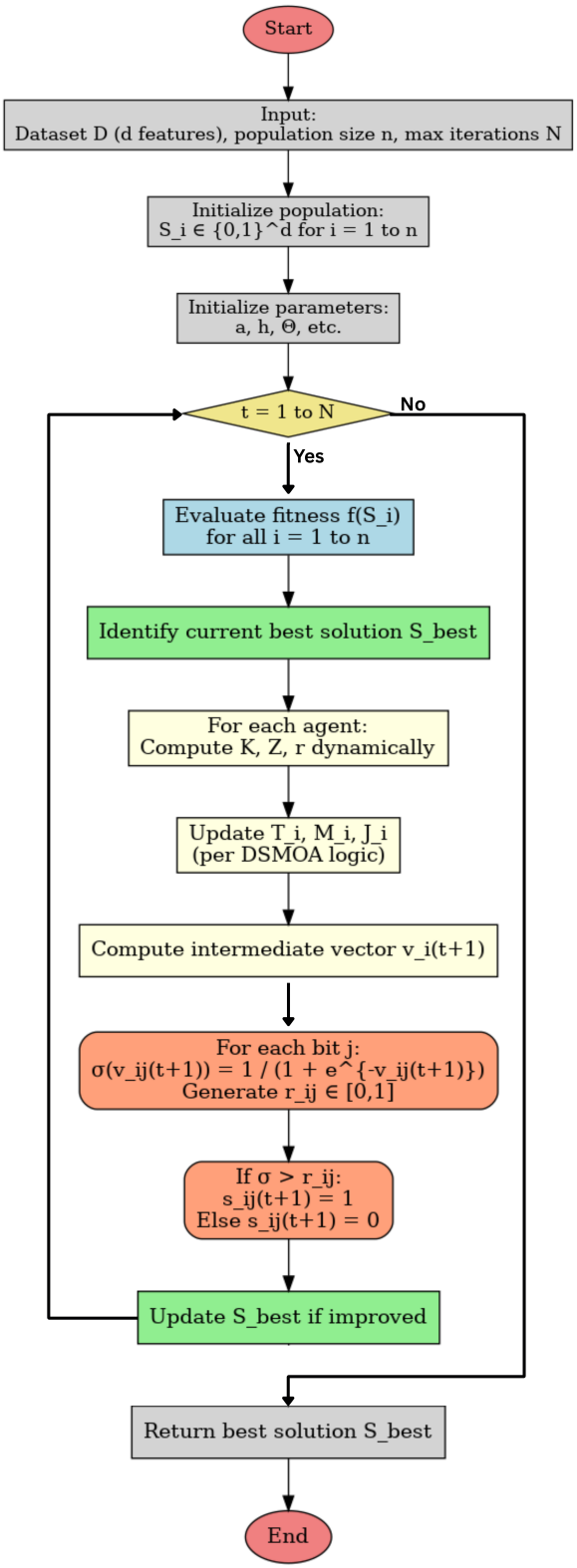

FIGURE 2: Flowchart of the proposed Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA).

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

where $\vec{T}$, $\vec{M}$, and $\vec{J}$ are computed using the DSMOA rules, and K, Z, and r are dynamic coefficients adjusted in each iteration based on stochastic parameters and progress.

This vector is then passed through a sigmoid-based transfer function to convert it into a binary solution:

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

where $r_{ij} \sim \mathcal{U}(0,1)$ and $\sigma(v) = \frac{1}{1+e^{-v}}$. This conversion controls the stochasticity of binary transitions and balances exploration and exploitation within a binary decision framework.

The three main behavior phases of DBSMOA are implemented with this formulation:

- **Exploration:** Uses stochastic updates with higher motion amplitudes to traverse the binary search space broadly.
- **Exploitation:** Applies focused movement near promising regions by tuning down randomness and increasing convergence pressure.
- **Elimination:** Introduces disruptive variations in stagnant regions to maintain diversity and avoid premature convergence.

These behaviors are dynamically regulated based on agent performance and population feedback, ensuring adaptability to various data characteristics.

3) Binary Transfer Mechanism

The binary conversion in DBSMOA relies on a sigmoid function to map continuous outputs to binary decisions:

$$s_{ij}(t+1) = \begin{cases} 1, & \text{if } \sigma(v_{ij}(t+1)) > r_{ij}, \\ 0, & \text{otherwise}, \end{cases} \quad \sigma(v) = \frac{1}{1+e^{-v}}, \quad (11)$$

where $r_{ij} \sim \mathcal{U}(0,1)$ is a uniformly distributed random number. This probabilistic thresholding enables smooth transitions and fine-grained control over binary decisions.

Alternative transfer functions (e.g., V-shaped, S-shaped) and adaptive thresholding strategies may be explored to enhance search dynamics depending on the problem context.

4) Feature Selection Representation

In binary feature selection, a candidate solution represents selected and unselected features. The binary vector clearly encodes this mapping, as shown in the example below:

This representation enables DBSMOA to efficiently search for feature subsets that improve classifier performance and reduce dimensionality.

5) Fitness Function

The fitness function in DBSMOA balances classification error with feature count using a weighted sum:

$$f(\vec{S}) = \alpha \cdot E(\vec{S}) + (1-\alpha) \cdot \frac{|\vec{S}|}{d}, \quad (12)$$

Algorithm 3 Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA)

Input: Dataset D with d features, population size n, max 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 parameters: a, h, Θ, etc.
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 solution $\vec{S}_{\text{best}}$
  for each agent $i = 1$ to n **do**
    Compute K, Z, r dynamically
    Update $\vec{T}_i$, $\vec{M}_i$, $\vec{J}_i$ per DSMOA logic
    Compute intermediate vector $\vec{v}_i(t+1)$
    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**
        Set $s_{ij}(t+1) = 1$
      else
        Set $s_{ij}(t+1) = 0$
      end if
    end for
  end for
  Update $\vec{S}_{\text{best}}$ if improved
end for
return $\vec{S}_{\text{best}}$

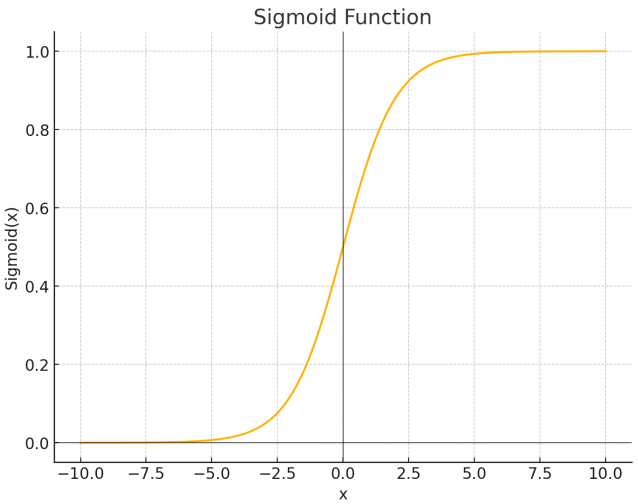

FIGURE 3: Sigmoid function curve illustrating the transformation from input x to bounded output $\sigma(x)$.

where:

- $E(\vec{S})$ is the classification error (e.g., $1 -$ accuracy) of the selected subset,

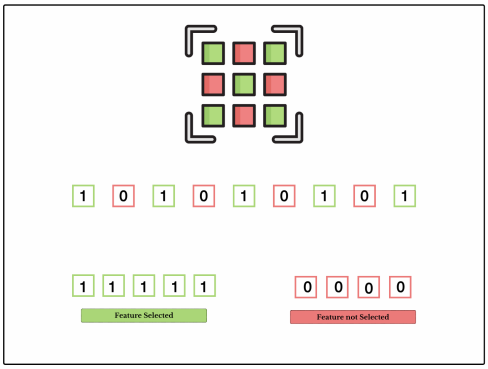

FIGURE 4: Binary representation of feature selection where green boxes represent selected features and red boxes indicate features that are not selected.

- $|\vec{S}|$ is the number of selected features,
- d is the total number of features,
- $\alpha \in [0,1]$ is a user-defined weight balancing accuracy and compactness.

Values of α near 1 emphasize accuracy, while lower values promote dimensionality reduction. This formulation offers flexible control over optimization goals and can be extended to multi-objective models if required.

6) 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:

- 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 3 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.

As shown in Table 3, 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 5 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.

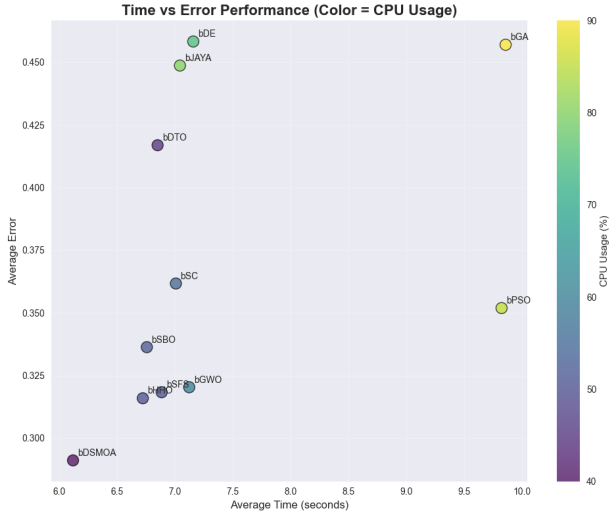

FIGURE 5: Time vs. Error Performance plot for all algorithms. Point color intensity encodes CPU usage.

Figure 6 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 4, 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.

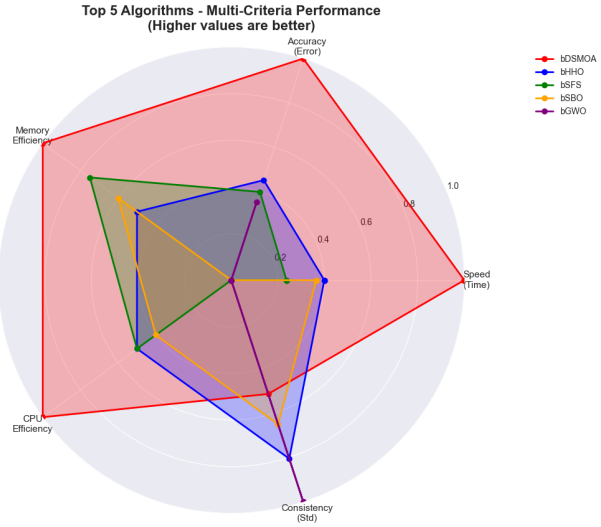

FIGURE 6: Radar plot comparing the top 5 algorithms across five normalized criteria.

Figure 7 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.

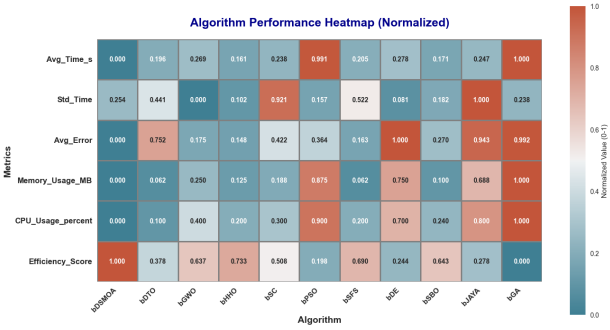

FIGURE 7: Heatmap of normalized performance metrics across all algorithms.

Figure 8 ranks the algorithms across various performance metrics. DBSMOA consistently achieves the best rankings in terms of time, error, memory, CPU usage, and 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.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

This section presents a comprehensive evaluation of the proposed Dynamic Binary Swordfish Movement Optimization

TABLE 3: 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

TABLE 4: 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 8: Bar charts ranking all algorithms based on time, error, consistency, memory, CPU usage, and overall efficiency.

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.

A. EXPERIMENTAL SETUP

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 5.

2) The Used Benchmark Algorithms

To comprehensively evaluate the performance of the proposed Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA), a set of widely recognized and well-established binary metaheuristic algorithms was selected as benchmarks. These algorithms span a diverse set of 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)** – A nature-inspired algorithm simulating avian flight dynamics [77].

TABLE 5: Summary of Benchmark Datasets Used to Evaluate DBSMOA

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

- **Binary Grey Wolf Optimizer (bGWO)** – Models the social hierarchy and hunting behavior of grey wolves [78].
- **Binary Harris Hawks Optimization (bHHO)** – Mimics the cooperative and surprise attack strategy of Harris hawks [79].
- **Binary Sine Cosine Algorithm (bSC)** – Uses a mathematical model of sine and cosine oscillations to navigate the search space [80].
- **Binary Particle Swarm Optimization (bPSO)** – Inspired by social behavior in bird flocking and fish schooling, adapted for binary spaces [81].
- **Binary Whale Optimization Algorithm (bWAO)** – Emulates the bubble-net feeding strategy of humpback whales [82].
- **Binary Stochastic Fractal Search (bSFS)** – A binary adaptation of the Stochastic Fractal Search algorithm, which leverages the principles of fractal propagation and diffusion for global and local search in high-dimensional binary spaces [83].
- **Binary Differential Evolution (bDE)** – An evolutionary algorithm modified to operate in discrete binary domains [84].
- **Binary Satin Bowerbird Optimizer (bSBO)** – Inspired by the courtship and construction behaviors of bowerbirds [85].
- **Binary JAYA Algorithm (bJAYA)** – A parameter-less optimizer that always aims to move toward the best and away from the worst solutions [86].
- **Binary Firefly Algorithm (bFA)** – Based on the flashing behavior of fireflies used for attraction and communication [87].
- **Binary Genetic Algorithm (bGA)** – A classical evolutionary algorithm using crossover and mutation tailored for binary strings [88].

These algorithms were selected due to their proven relevance in binary optimization and feature selection research, and they serve as a robust baseline to compare the effectiveness, efficiency, and reliability of DBSMOA. Uniform parameter settings, fitness evaluation criteria, and stopping conditions were maintained across all methods to ensure fair and consistent comparisons.

3) Parameter Settings

To ensure a fair and consistent comparison, all algorithms were executed using the same general configuration parameters, including a population size of 30, a maximum of 100 iterations, and 30 independent runs. Each algorithm was applied to the same benchmark datasets using a fixed number of agents (20), and classification accuracy was evaluated using a k-Nearest Neighbors (k-NN) classifier with $k = 5$.

Algorithm-specific parameters were carefully selected based on the original literature or standard settings known to yield optimal performance. These include internal coefficients, randomization factors, binary transfer functions, mutation rates, and control constants relevant to each metaheuristic. For newly adapted or hybrid algorithms, such as the proposed Dynamic Binary Swordfish Movement Optimization Algorithm (bDSMOA), original dynamic parameters inherited from the continuous Swordfish Movement Optimization Algorithm (SMOA) were retained and adapted for binary behavior.

A comprehensive summary of all parameter settings used in the experiments is provided in Table 6. These values were kept constant across all datasets to preserve comparability and minimize tuning bias.

TABLE 6: Parameter Settings for All Algorithms

All Algorithms	Parameter
Population size	30
Number of iterations	100
Number of runs	30
Number of agents	20
bDSMOA	a: [0, 1] h: Motion intensity constant Θ: Angular parameter K: $\left(1+\frac{(\text{iteration})^2}{\text{iteration}}\right)^2$ Z: $\left(X+\frac{2\cdot(\text{iteration})^2}{N}\right)^2$ r: $r=\frac{h\cdot\cos(X)}{1-\cos(X)}$
bDTO	C_1: $2c\cdot r_1 - c$, where $c=2\left(1-\left(\frac{t}{T_{\max}}\right)^2\right)$ C_2: $2r_1$ C_3: Inertia weight for velocity C_4, C_5: Acceleration constants r_1, r_2, R: Random numbers in $[0, 1]$
bHHO	Jump Strength (J): random value in [0, 2] Escaping Energy (E): $E=2E_0(1-\frac{t}{T})$ E_0: random in $[-1, 1]$
bGWO	a: 2 to 0
bPSO	C_1 (Cognitive Constant): 2 C_2 (Social Constant): 2 W (Inertia Constant): 0.3
bWOA	b (Spiral Shape): Linearly decreased from 2 to 0
bSFS	r: [0, 1]
bDE	Crossover Probability: 0.5 Mutation Factor: 0.5
bSBO	r_2, r_3, r_4: [0, 1]
bJAYA	Variable Range (x_i): [-100, 100] Random Numbers (r_1, r_2): [0, 1]
bFA	Wormhole Existence Probability: [0.2, 1] Step Size: 0.94
bGA	Mutation Probability: 0.05 Crossover: 0.02 # Fireflies: 10
bSCA	Mutation Ratio: 0.1 Crossover: 0.9

4) Evaluation Metrics

To rigorously evaluate the performance of the proposed DBSMOA algorithm, several statistical and optimization-based metrics are employed [89], [90]. Table 7 summarizes the definition, purpose, and equation for each metric used across M independent runs.

B. 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 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 8, 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 evaluate the performance of various metaheuristic algorithms in the context of feature selection, we performed a kernel density estimation (KDE) analysis of the feature selection scores generated by each algorithm. This analysis allows us to observe the distribution and concentration of scores across different methods, providing insight into their relative effectiveness and consistency. As illustrated in Figure 9, the KDE plots reveal distinctive density patterns for each algorithm, highlighting differences in central tendencies and variability.

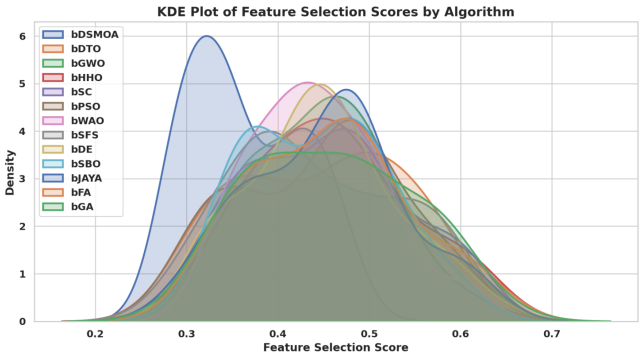

FIGURE 9: KDE plot of feature selection scores for various algorithms including DBSMOA, bDTO, bGWO, bHHO, bSC, bPSO, bWAO, bSFS, bDE, bSBO, bJAYA, bFA, and bGA. The density curves allow visual comparison of score distributions across methods.

To complement the density-based analysis of feature selection performance, a swarm plot was used to provide a detailed, point-wise visualization of the individual feature selection scores produced by each algorithm. This approach enables a clear comparison of score dispersion and potential outliers, providing insight into the stability and consistency of each method's performance. As shown in Figure 10, the swarm plot highlights the variability within and between algorithms, revealing how tightly the scores cluster and indicating trends or anomalies across different techniques.

To gain a comprehensive understanding of both the distribution and spread of feature selection scores, we employed a violin plot visualization. This method combines the benefits of boxplots and kernel density plots, enabling the simultaneous display of central tendency (via quartiles) and data distribution shape. As depicted in Figure 11, each violin represents an algorithm's score distribution, allowing for easy comparison of variability, median values, and score density across different feature selection methods.

TABLE 7: Summary of evaluation metrics used for DBSMOA assessment.

Metric	Description	Equation
Average Classification Error (AvgError)	Average misclassification rate over all samples and runs. Lower is better.	$\text{AvgError} = 1 - \frac{1}{M}\sum_{j=1}^{M}\left(\frac{1}{N}\sum_{i=1}^{N}\text{Match}(C_i, L_i)\right)$
Average Number of Selected Features (AvgSelectSize)	Average ratio of selected features to total features, measuring reduction effectiveness.	$\text{AvgSelectSize} = \frac{1}{M}\sum_{j=1}^{M}\frac{\text{size}(g_j^*)}{D}$
Average Fitness (AvgFitness)	Mean fitness score over all runs, indicating optimization performance.	$\text{AvgFitness} = \frac{1}{M}\sum_{j=1}^{M} g_j^*$
Best Fitness (BestFn)	Best (minimum) fitness value found among all runs.	$\text{BestFn} = \min_{1\le j\le M} f(g_j^*)$
Worst Fitness (WorstFn)	Worst (maximum) fitness value across all runs.	$\text{WorstFn} = \max_{1\le j\le M} f(g_j^*)$
Standard Deviation of Fitness (SD)	Measures variability in fitness values, reflecting solution stability.	$\text{SD} = \sqrt{\frac{1}{M-1}\sum_{j=1}^{M}(g_j^* - \text{Mean})^2}$

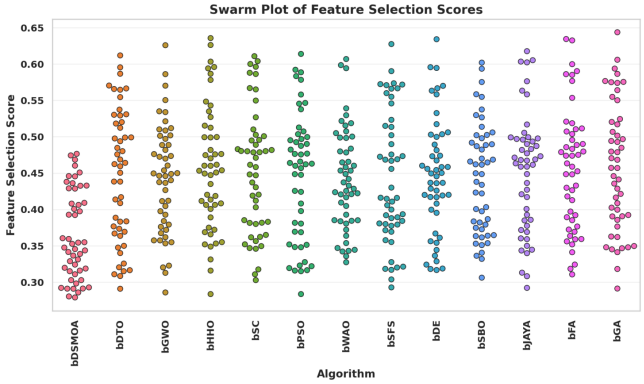

FIGURE 10: Swarm plot showing the distribution of feature selection scores for each algorithm. Each dot represents an individual run, enabling visual assessment of performance spread and consistency.

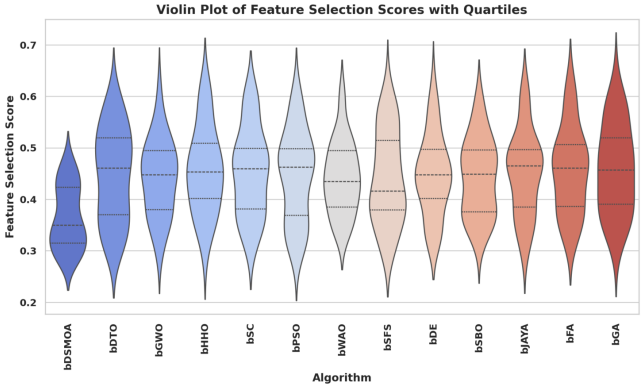

FIGURE 11: Violin plot illustrating the distribution of feature selection scores for each algorithm, overlaid with internal quartile markers. This visualization captures both central values and score variability.

The **average number of selected features** reflects the dimensionality reduction capability of each optimizer. DBSMOA achieved the smallest average subset size across *34 datasets*, reflecting its efficiency in filtering out redundant or irrelevant features, as shown in Table 9. For instance, in the *warpAR10P* dataset, DBSMOA selected only **26.9%** of features compared to bSFS (61.0%) and bDE (47.9%). Likewise, for the *Parkinsons* dataset, it achieved a substantial reduction at **34.5%**, which is far lower than the 58–79% ranges of competing methods. These results validate the algorithm's ability to conduct aggressive yet effective feature pruning.

To further understand the distributional behavior of feature selection scores across different algorithms, a cumulative distribution function (CDF) plot was utilized. The CDF provides a comprehensive view of how scores accumulate, highlighting differences in the proportion of low vs. high values and the overall statistical behavior of each method. As depicted in Figure 12, this visualization facilitates direct comparisons of algorithmic robustness and performance consistency by showing the cumulative probabilities over the score range.

To assess and compare the empirical performance of several algorithms, we applied kernel density estimation (KDE) to their performance distributions. This approach helps reveal not only the central tendencies but also the variability and modality of each algorithm's results. As illustrated in Fig-

TABLE 8: 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

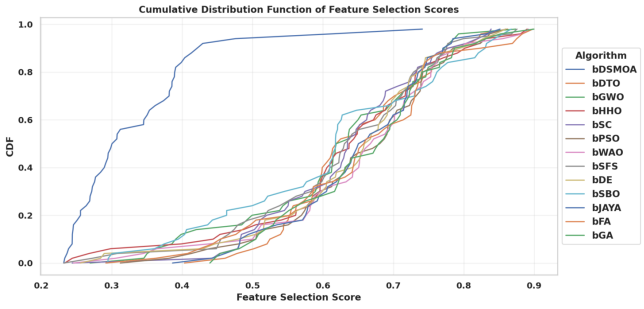

FIGURE 12: Cumulative Distribution Function (CDF) of feature selection scores for all evaluated algorithms. The plot demonstrates how feature scores accumulate, enabling insight into each algorithm's statistical behavior.

ure 13, the KDE plot provides a smoothed representation of the distribution of performance scores for the bSCWDTO, bDTO, bSC, and bPSO algorithms, enabling a visual assessment of how their outcomes differ in consistency and effectiveness.

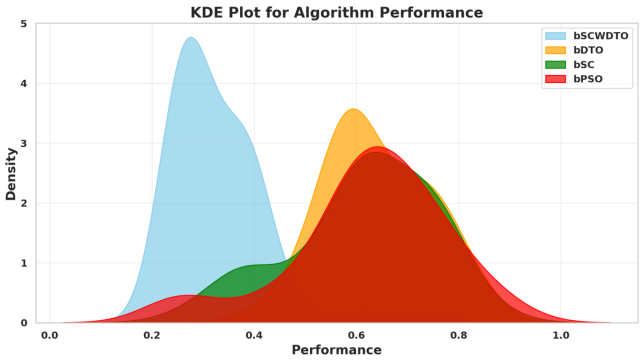

FIGURE 13: KDE plot showing the performance distributions of the bSCWDTO, bDTO, bSC, and bPSO algorithms. The density curves provide insight into the comparative performance behavior and variability of each method.

To provide a detailed view of the value distribution for

TABLE 9: 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

each algorithm, histograms with overlaid kernel density estimation (KDE) curves were generated. These plots allow for the examination of central tendency, skewness, and spread of performance or feature selection values for each method independently. As shown in Figure 14, each subplot corresponds to a specific algorithm, enabling side-by-side comparisons of frequency distributions and identifying unique characteristics or anomalies in their behavior.

The **average fitness** function combines classification accuracy with feature reduction into a unified scalar objective. DBSMOA yielded the lowest average fitness in the majority of cases, indicating its balanced optimization behavior, as shown in Table 10. For example, in the *HeartEW* dataset, DBSMOA attained an average fitness of **0.40204**, lower than bFA (0.53624) and bWAO (0.57514). A similar trend is observed in the *Kc2* dataset, where DBSMOA achieves **0.3660**, outperforming bGWO (0.3922) and bGA (0.5391). This reinforces DBSMOA's efficiency in navigating the search space toward Pareto-optimal solutions.

To investigate the relationship between the performance of the DBSMOA and bDTO algorithms, a bivariate density heatmap was generated. This type of visualization provides insight into how the results from the two algorithms co-vary, revealing regions of high and low joint density. Such analysis helps assess potential correlations or systematic performance similarities. As shown in Figure 15, the heatmap highlights areas of high sample concentration, illustrating the extent to which the outputs of DBSMOA and bDTO align.

To analyze the consistency and comparative progression of feature selection scores over multiple runs, a trend line plot was created for all evaluated algorithms. This visualization enables temporal insight into performance stability and variation, highlighting fluctuations and convergence behaviors. As shown in Figure 16, each line represents a specific algorithm, allowing for direct observation of score trends across multiple observations and identifying potential outliers or recurring patterns.

To explore the degree of similarity and linear association between the feature selection scores produced by different algorithms, a correlation heatmap was constructed. This vi-

TABLE 10: 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

sualization helps identify which algorithms tend to produce similar outcomes, thereby shedding light on potential redundancy or complementarity among methods. As illustrated in Figure 17, each cell in the matrix represents the Pearson correlation coefficient between a pair of algorithms, with warmer colors indicating stronger positive correlations and cooler tones reflecting weaker relationships.

The **best fitness** value measures the optimizer's most optimal solution across all independent trials. DBSMOA achieved the best fitness in numerous datasets as shown in Table 11. For instance, it reached the lowest score of **0.24861** on the *Breast cancer Coimbra* dataset, a performance unmatched by all other methods. In the *Mofn* dataset, it recorded **0.44146**, outperforming bPSO (0.64116) and bGA (0.48376). This demonstrates DBSMOA's powerful exploitation capability and its ability to converge to superior local optima.

To visually assess the pairwise relationships among the feature selection scores obtained from different algorithms, a pair plot was generated. This comprehensive grid of scatter plots and distributions enables the exploration of potential linear or nonlinear associations between algorithmic outputs while also capturing distributional properties along the diagonals. Additionally, color-coded annotations enhance interpretability by grouping data points based on score intervals. As shown in Figure 18, the pairplot reveals structural patterns and clustering tendencies across algorithms, which can be indicative of similarity in selection behavior or performance characteristics.

To provide a compact and comparative overview of the average feature selection scores achieved by each algorithm, a radar plot was constructed. This form of visualization is especially useful for assessing the relative performance of multiple methods across a common scale in a circular layout. As shown in Figure 19, the plot enables a quick visual comparison of algorithmic effectiveness, where longer radii correspond to higher average scores, thereby highlighting the leading and lagging methods in terms of overall feature selection quality.

Conversely, the **worst fitness** score highlights the least favorable solution discovered. An optimizer with lower worst-

TABLE 11: 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

case values indicates robustness and immunity to poor convergence scenarios. DBSMOA consistently maintained a better worst-case performance as shown in Table 12, such as **0.2678** on the *Ionosphere* dataset and **0.25085** on the *Fri_c0_1000_10* dataset — both significantly outperforming the respective worst scores of bPSO, bDE, and bFA. These outcomes emphasize the algorithm's resilience and controlled variance.

To uncover structural relationships and group similarities among the feature selection algorithms, hierarchical clustering was applied based on their pairwise correlation coefficients. This method organizes the algorithms into a tree-like structure, grouping those with the most similar score behaviors at lower linkage distances. As illustrated in Figure 20, the dendrogram provides a visual summary of the similarity hierarchy, enabling the identification of clusters of methods that exhibit consistent correlation patterns in their outputs.

To provide a comprehensive summary of the distributional properties of feature selection scores across all algorithms, box plots with overlaid swarm plots and statistical annotations were generated. This visualization captures both the central tendency and dispersion of scores while simultaneously displaying individual data points for clarity. Mean and standard deviation indicators further enhance interpretability by quantifying average performance and variability. As illustrated in Figure 21, each subplot corresponds to a specific algorithm, allowing for a side-by-side comparison of distribution shapes, outlier prevalence, and consistency.

To facilitate a focused inspection of the feature selection score distributions for each algorithm independently, individual boxplots were constructed and organized in a grid layout. This visualization format allows for a direct comparison of central tendency, spread, and potential outliers within each method's performance. By isolating each algorithm into its panel, Figure 22 provides a transparent and interpretable view of variability and skewness, allowing for an objective assessment of score reliability and dispersion across algorithms.

Finally, the **standard deviation of fitness values** represents the optimizer's stability across independent executions.

TABLE 12: 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

Lower values suggest consistent performance regardless of initialization or stochastic effects. In more than *30 datasets*, DBSMOA achieved the lowest deviations shown in Table 13. For instance, in the *Seeds* dataset, it attained a deviation of only **0.17643**, in contrast to bHHO (0.2033) and bGA (0.34803). Likewise, for the *Estimation of Obesity* dataset, DBSMOA reported the most stable behavior with **0.17659**, affirming its superior reliability in uncertain environments.

To further investigate the distributional characteristics of feature selection scores across algorithms, a violin plot was employed. This type of plot visualizes both the probability density and the summary statistics (e.g., median and quartiles), offering a rich representation of the spread and modality of score distributions. As illustrated in Figure 23, each violin corresponds to a specific algorithm, allowing for comparative analysis of score concentration, variability, and symmetry across methods.

To analyze the score distribution of each algorithm in a more detailed and isolated manner, individual violin plots with kernel density estimation (KDE) overlays were constructed. These plots not only show the density of values along the performance axis but also incorporate rug plots to display the underlying data points. As illustrated in Figure 24, each subplot highlights the shape and spread of the feature selection scores for a specific algorithm, facilitating a granular comparison of distribution characteristics such as modality, symmetry, and kurtosis.

Overall, these performance metrics demonstrate the clear advantage of DBSMOA over classical and state-of-the-art binary optimizers, particularly in maintaining a balanced trade-off between compactness of selected features and high predictive performance across a broad spectrum of datasets and problem domains.

C. STATISTICAL ANALYSIS

To evaluate the statistical significance of the differences in performance among the compared binary optimization algorithms, a comprehensive pairwise analysis of p-values was conducted using multiple benchmark datasets. The results, summarized in Table 14, present the computed p-values

TABLE 13: 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

for each algorithm pair across all datasets based on their classification performance.

This matrix of significance levels enables an objective comparison of whether the observed differences between any two optimizers are statistically meaningful or potentially due to chance. The lower the p-value, the more substantial the evidence against the null hypothesis of no difference between the two optimizers.

As shown in Table 14, the majority of p-values across the optimizer pairs are substantially below the standard threshold of 0.05. This confirms that the differences among the optimizers are not only evident but also statistically robust. For instance, on datasets like *WineEW*, *warpAR10P*, and *Robot-failures-lp1*, the vast majority of comparisons yield extremely small p-values (e.g., in the order of 10^{-7} or lower), indicating highly significant differences in performance across multiple optimizers.

To provide a holistic comparison of the performance characteristics across all feature selection algorithms, a stacked bar chart was utilized. This visualization consolidates several key metrics—average error, average selected feature size, average fitness, best fitness, worst fitness, and fitness standard deviation—into a single, unified view for each algorithm. As shown in Figure 25, the stacking of these metrics enables a comparative assessment of overall behavior, highlighting differences in selection quality, consistency, and accuracy.

To enable a multi-criteria comparison of feature selection algorithms, a radar plot was constructed using six normalized performance metrics: average error, average selected size, average fitness, best fitness, worst fitness, and standard deviation of fitness. This type of visualization is particularly useful for identifying balanced performance and outliers across multiple dimensions simultaneously. As shown in Figure 26, each polygon represents a single algorithm's profile, facilitating an intuitive assessment of strengths and trade-offs among the competing methods.

To simultaneously examine the performance of multiple feature selection algorithms across a diverse set of evaluation metrics, a parallel coordinates plot was constructed. This visualization facilitates the comparison of algorithmic behavior

TABLE 14: 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**

by representing each algorithm as a line traversing multiple axes—each corresponding to a normalized performance metric. As illustrated in Figure 27, this plot highlights how algorithms trade off among key criteria, such as average error, selected feature size, fitness values, and fitness variability. Crossing lines indicate differing strengths and weaknesses, enabling the identification of balanced versus specialized methods.

To evaluate the degree of similarity in performance profiles among different feature selection algorithms, a correlation heatmap was generated based on the normalized metric vectors of each algorithm. This visualization captures the pairwise relationships between algorithms in terms of overall performance behavior across multiple criteria. As shown in Figure 28, the color intensity represents the strength of the correlation, with values close to 1.0 indicating high similarity. The heatmap reveals clusters of algorithms with highly aligned metric patterns, highlighting redundancies or consistent performance trends.

To rigorously evaluate the performance variations among the binary optimizers on the WDBC dataset, we conducted a comprehensive statistical analysis using three tools: one-way ANOVA, one-sample t-tests, and descriptive statistics.

The one-way ANOVA test, summarized in Table 15, was applied to examine whether significant differences exist among the means of the competing optimizers. The results indicate a highly significant difference, with an F-statistic of 411.0 at $F(12, 247)$ and a p-value less than 0.0001. This suggests that at least one optimizer shows a statistically distinct performance level on the WDBC dataset, justifying further pairwise evaluations.

Following the ANOVA, a one-sample t-test was conducted to assess whether the mean performance of each optimizer significantly differs from a theoretical baseline of zero. As detailed in Table 16, all optimizers yielded statistically significant deviations ($p < 0.0001$), confirming their effectiveness on the dataset.

Among them, DBSMOA recorded a mean classification

TABLE 15: ANOVA results for WDBC dataset

ANOVA table	SS	DF	MS	F (DFn, DFd)	P value
Treatment (between columns)	0.7467	12	0.06222	F (12, 247) = 411.0	P < 0.0001
Residual (within columns)	0.03739	247	0.0001514		
Total	0.7841	259			

error of 0.342 with a 95% confidence interval of [0.3405, 0.3435], which is both low and narrow, demonstrating its precision and stability. Its t-statistic reached 486.6 with an R-squared of 0.9999, indicating an extremely strong effect. In contrast, bWAO exhibited higher variance (standard deviation = 0.01921), reflecting greater inconsistency. These results collectively highlight DBSMOA's statistical advantage in stability and performance accuracy on the WDBC dataset.

Table 17 provides comprehensive descriptive measures, highlighting the central tendency, spread, and distribution characteristics for each optimizer. Notably, DBSMOA 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.

To evaluate the classification performance of each algorithm on the WDBC dataset, the average error rates were compared and visualized using a violin plot. As shown in Figure 29, the distribution and central tendency of the objective function values are illustrated for each algorithm. The width of each violin indicates the density of data points, helping to distinguish the consistency and variance in performance. Notably, the DBSMOA and bPSO algorithms demonstrate lower average error values compared to others, highlighting their potential effectiveness on this dataset.

To visualize the statistical significance and magnitude of differences among algorithms on the WDBC dataset, a heat map was generated based on the results of an ordinary one-way analysis of variance (ANOVA). As shown in Figure 30, the color gradients indicate the direction and strength of deviation in performance across selected features. Positive differences are represented in red shades, while green shades denote negative deviations, facilitating quick identification of comparative behavior among the DBSMOA, bGWO, bSC, bWAO, bDE, bJAYA, and bGA algorithms.

To validate the assumption of homoscedasticity—constant variance of residuals—in the analysis of variance for the WDBC dataset, a homoscedasticity plot was generated. Figure 31 presents the absolute residuals plotted against the predicted values. The overall distribution of the residuals appears randomly scattered without a clear pattern, supporting the assumption of homoscedasticity. This lends credibility to the reliability of the applied linear model for inferential conclusions.

To assess the normality assumption of the residuals in the analysis of the WDBC dataset, a Quantile-Quantile (QQ) plot was employed. As depicted in Figure 32, the predicted residuals are plotted against the actual residuals alongside a reference line representing the expected normal distribution. While minor deviations from the line are visible, especially in the tails, the majority of points lie reasonably close to the reference line, suggesting that the residuals approximately follow a normal distribution.

To assess the significance of the differences in average classification error rates obtained by various optimization algorithms on the Vehicle dataset, a comprehensive statistical analysis was conducted. This included one-way Analysis of Variance (ANOVA), one-sample t-tests, and descriptive statistics. The one-way ANOVA results (as detailed in Table 18) yielded an F-statistic of 618.3 with degrees of freedom $(12, 247)$ and a P-value of $P < 0.0001$. This clearly indicates that statistically significant differences exist among the means of the tested algorithms. The between-group sum of squares (SS) was 0.7097, while the within-group residual was 0.02363. These values support the robustness of the model's explanatory power.

Furthermore, the one-sample t-tests were carried out for each optimizer with a null hypothesis of zero mean classification error. As observed in Table 19, all optimizers, including DBSMOA, bDTO, bGWO, bHHO, and others, exhibited t-statistics with corresponding P-values less than 0.0001, confirming the significance of their performances. Notably, DBSMOA achieved a mean error of 0.2978 with a minimal standard deviation of 0.001814, and an exceptionally tight confidence interval of [0.2970, 0.2987]. This is complemented by an R^2 value of 1.0, indicating a perfect effect explanation in this context.

Descriptive statistics summarized in Table 20 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.

To evaluate and compare the performance of different feature selection algorithms on the Vehicle dataset, a violin-style distribution plot of the objective function values was employed. This visualization illustrates both the central tendency and the variability in optimization outcomes for each method, with distinctive markers and colors facilitating identification. As illustrated in Figure 33, the distribution width reflects performance consistency, while position along the vertical axis indicates effectiveness. This enables a clear visual comparison of how each algorithm performs in minimizing the objective function on this specific dataset.

To investigate the statistical significance of performance

TABLE 16: One-sample t-test results for WDBC dataset

	DBSMOA	bDTO	bGWO	bHHO	bSC	bPSO	bWAO	bSFS	bDE	bSBO	bJAYA	bFA	bGA
Theoretical mean	0	0	0	0	0	0	0	0	0	0	0	0	0
Actual 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
Number of values	20	20	20	20	20	20	20	20	20	20	20	20	20
t, df	t=486.6, df=19	t=381.3, df=19	t=399.9, df=19	**t=701.4, df=19**	t=334.3, df=19	t=437.7, df=19	t=117.8, df=19	t=279.7, df=19	t=500.0, df=19	t=159.7, df=19	t=199.6, df=19	t=71.32, df=19	t=104.3, df=19
One sample t test													
P value	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001
Significant?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Discrepancy													
Discrepancy	**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
SD	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
SEM	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
95% CI	[0.3405, 0.3435]	[0.4954, 0.5009]	[0.3762, 0.3802]	**[0.4744, 0.4773]**	[0.4104, 0.4156]	[0.3669, 0.3704]	[0.4972, 0.5151]	[0.4090, 0.4152]	[0.4649, 0.4688]	[0.3833, 0.3935]	[0.4919, 0.5023]	[0.4589, 0.4867]	[0.3964, 0.4126]
R^2	0.9999	0.9999	0.9999	**1**	0.9998	0.9999	0.9986	0.9998	0.9999	0.9993	0.9995	0.9963	0.9983

TABLE 17: 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

TABLE 18: ANOVA results for vehicle dataset

ANOVA table	SS	DF	MS	F (DFn, DFd)	P value
Treatment (between columns)	0.7097	12	0.05914	F (12, 247) = 618.3	P<0.0001
Residual (within columns)	0.02363	247	0.00009565		
Total	0.7333	259			

TABLE 19: One-sample t-test results for vehicle dataset

	DBSMOA	bDTO	bGWO	bHHO	bSC	bPSO	bWAO	bSFS	bDE	bSBO	bJAYA	bFA	bGA
Theoretical mean	0	0	0	0	0	0	0	0	0	0	0	0	0
Actual 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
Number of values	20	20	20	20	20	20	20	20	20	20	20	20	20
One sample t test													
t, df	**t=734.1, df=19**	t=639.7, df=19	t=202.6, df=19	t=240.0, df=19	t=134.2, df=19	t=139.0, df=19	t=157.6, df=19	t=257.8, df=19	t=291.7, df=19	t=125.7, df=19	t=112.8, df=19	t=154.8, df=19	t=144.9, df=19
P value	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001
Significant?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
discrepancy													
Discrepancy	**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
SD	**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
SEM	**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
95% CI	**[0.2970 , 0.2987]**	[0.4626 , 0.4656]	[0.3641 , 0.3717]	[0.3651 , 0.3715]	[0.3565 , 0.3678]	[0.3197 , 0.3295]	[0.3245 , 0.3332]	[0.4496 , 0.4570]	[0.4525 , 0.4590]	[0.3621 , 0.3743]	[0.3519 , 0.3652]	[0.3555 , 0.3652]	[0.4311 , 0.4437]
R^2	**1**	**1**	0.9995	0.9997	0.9989	0.999	0.9992	0.9997	0.9998	c 0.9988	0.9985	0.9992	0.9991

TABLE 20: 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

differences among feature selection algorithms applied to the Vehicle dataset, an ordinary one-way ANOVA test was conducted. The resulting deviations across samples and methods were visualized using a heatmap, where each cell represents the magnitude and direction of deviation from the mean. In Figure 34, green shades indicate negative deviations (better than the group mean), while red shades reflect positive deviations (worse performance). This heatmap allows for the detection of patterns in performance consistency and outliers across individual data points.

To assess whether the residuals from the one-way ANOVA applied to the Vehicle dataset satisfy the assumption of homoscedasticity, a homoscedasticity plot was generated. This plot visualizes the absolute residuals against the predicted values, serving as a diagnostic tool to check for constant variance across the range of the regression. As shown in Figure 35, the spread of the residuals appears relatively uniform across the predicted values, which supports the assumption of equal variance and strengthens the validity of the ANOVA results.

To verify the normality assumption of residuals in the one-way ANOVA performed on the Vehicle dataset, a Q-Q

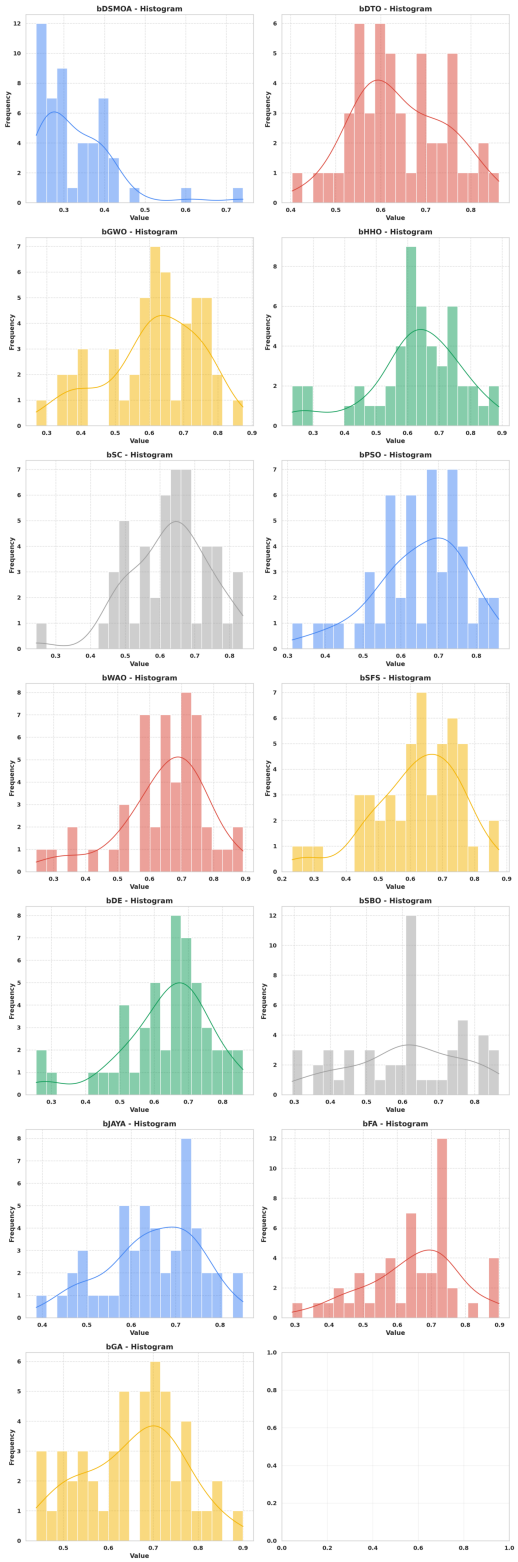

FIGURE 14: Histogram plots with KDE overlays for individual algorithms. Each subplot displays the frequency distribution of values obtained by a specific algorithm, providing insights into their statistical tendencies.

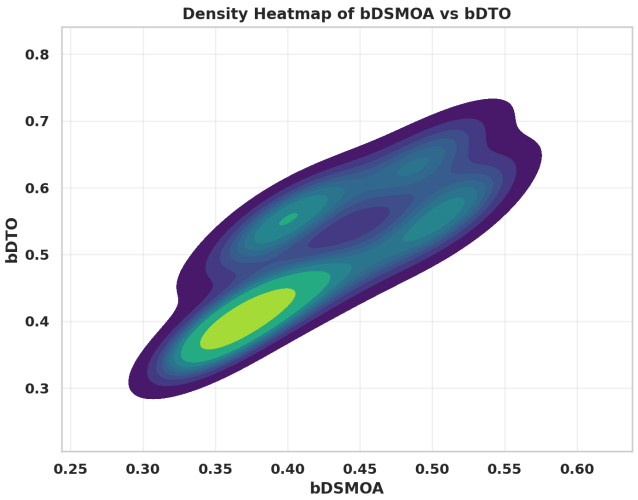

FIGURE 15: Density heatmap comparing the joint distribution of feature selection scores between DBSMOA and bDTO. Darker regions indicate lower density, while brighter areas show where the results of the two algorithms overlap most frequently.

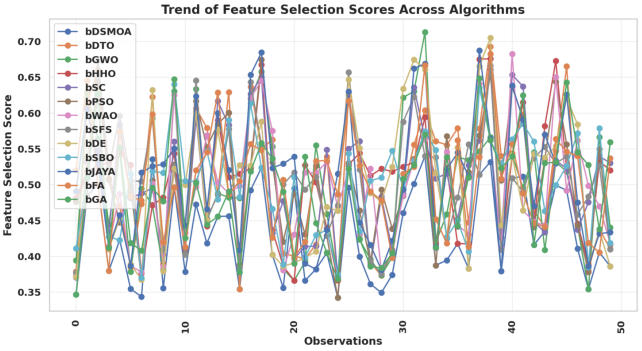

FIGURE 16: Line plot illustrating the trend of feature selection scores across multiple observations for each algorithm. This visual aid helps assess consistency, variability, and overall score progression.

(quantile-quantile) plot was constructed. This diagnostic tool compares the distribution of actual residuals to a theoretical normal distribution. As visualized in Figure 36, data points aligning closely along the red reference line suggest that the residuals are approximately normally distributed. Deviations from the line, particularly in the tails, can indicate skewness or kurtosis that may affect the robustness of ANOVA conclusions.

These results provide strong statistical evidence that the tested optimizers, particularly DBSMOA, demonstrate significantly distinct performance levels when applied to the Vehicle dataset.

To validate the differences in performance among the evaluated binary optimizers on the **Page Blocks** dataset, a comprehensive statistical evaluation was conducted. This included a one-way ANOVA test, one-sample t-tests, and de-

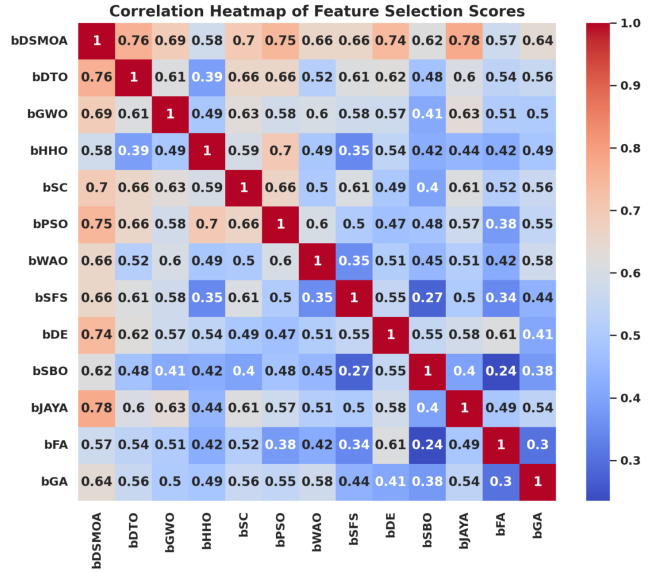

FIGURE 17: Correlation heatmap of feature selection scores between algorithms. The matrix reveals the strength of pairwise correlations, offering insights into algorithmic similarity in selection behavior.

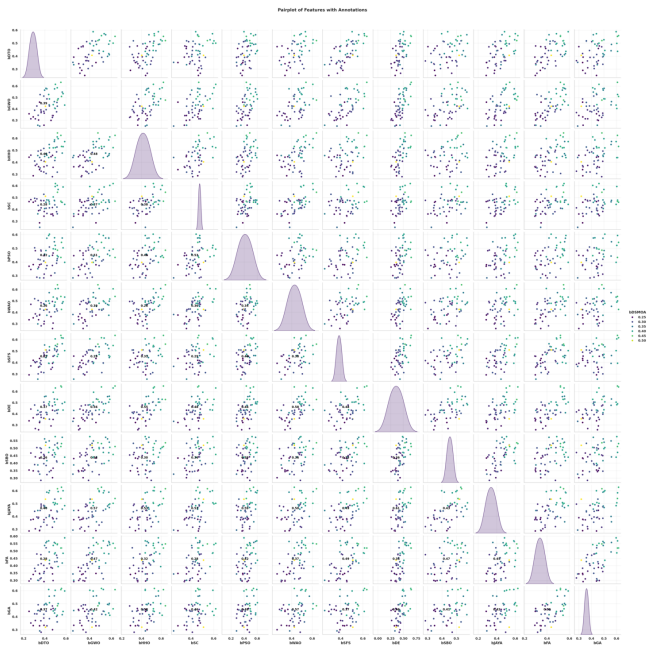

FIGURE 18: Pairplot showing scatter and distribution plots of feature selection scores across algorithms. This grid allows visual comparison of pairwise relationships and score distributions, with color-coded annotations representing different score ranges.

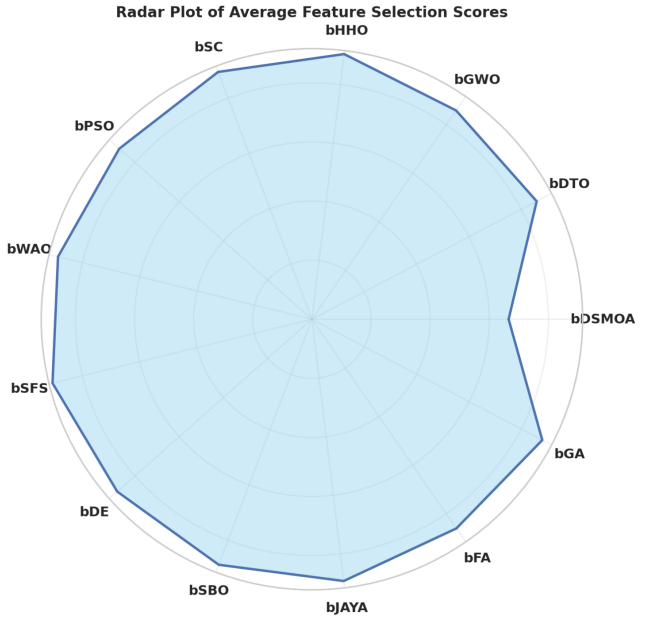

FIGURE 19: Radar plot depicting the average feature selection scores for each algorithm. This layout facilitates an intuitive comparison of performance across all methods within a unified radial space.

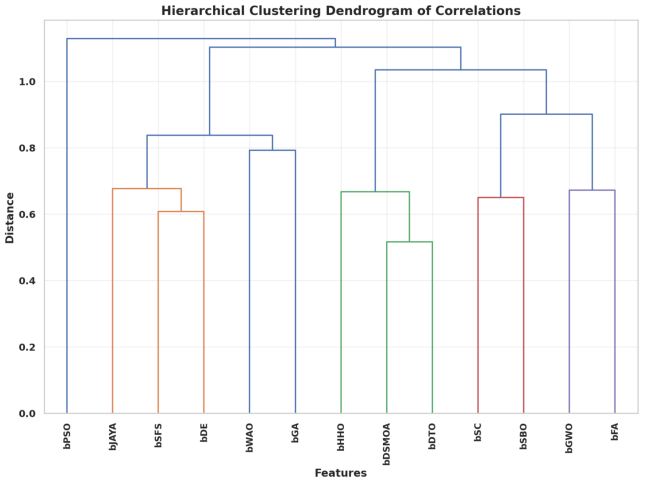

FIGURE 20: Hierarchical clustering dendrogram of correlations between feature selection algorithms. Algorithms that behave similarly in terms of output correlation are grouped together more closely.

scriptive statistics. These analyses aim to identify whether the observed performance differences are statistically significant and to characterize the distribution of the results.

The one-way ANOVA was applied to assess whether there were significant differences in the mean error values among the optimizers. As shown in Table 21, the test yielded an F-statistic of 559.8 with a p-value less than 0.0001, indicating a statistically significant difference between at least one pair of optimizer means. This confirms that the performance variations are not due to random chance and merit deeper inspection.

To further understand individual optimizer performance, one-sample t-tests were conducted, comparing each opti-

TABLE 21: ANOVA results for Page Blocks dataset

ANOVA table	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			

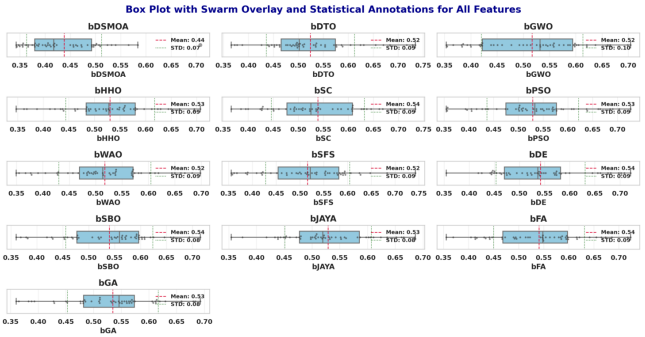

FIGURE 21: Box plots with swarm overlays for each algorithm. The plots show score distributions, individual data points, mean (dashed red line), and standard deviation bounds (dotted green lines) to support comparative analysis of performance variability.

mizer's average error against a null hypothesis mean of zero. Table 22 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 23, **DBSMOA** achieved the lowest 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 assess the performance of various feature selection algorithms on the Page Blocks dataset, a customized violin plot was constructed to display the distribution of objective function values. Each algorithm is represented with a unique marker and color to enhance visual differentiation, while the spread of the shapes indicates variability and central tendencies in the results. As depicted in Figure 37, this figure allows for a clear comparison of optimization quality across methods, highlighting both consistency and effectiveness in minimizing the objective function.

To statistically assess the variance in performance across different feature selection algorithms on the Page Blocks dataset, a one-way analysis of variance (ANOVA) was conducted. The resulting differences in means were visualized using a heatmap, where each cell represents the level of deviation for a given instance and algorithm. The color intensity encodes the direction and magnitude of the variation, with green indicating negative deviations and red indicating positive ones. As shown in Figure 38, this visualization enables quick identification of patterns, anomalies, and consistency (or lack thereof) in algorithmic behavior across data points.

To validate one of the key assumptions of the ordinary one-way ANOVA—namely, the assumption of homoscedasticity (equal variance of residuals across levels of the predictor)—a homoscedasticity plot was generated. This plot displays the absolute residuals against the predicted values, enabling a visual inspection of variance consistency. As illustrated in Figure 39, the absence of a clear funnel shape or systematic spread in the residuals supports the homogeneity of variance assumption required for valid ANOVA interpretation.

To evaluate whether the residuals from the one-way ANOVA applied to the Page Blocks dataset follow a normal distribution—a key assumption for the validity of ANOVA inference—a Q-Q (quantile-quantile) plot was generated. This plot compares the actual residuals to the theoretical quantiles of a normal distribution. As depicted in Figure 40, deviations from the red reference line indicate departures from normality. While moderate curvature may suggest slight non-normality, the plot overall helps visually diagnose whether the residuals meet the ANOVA's assumption of normality.

Overall, this tri-layered analysis confirms that optimizer selection significantly influences performance and that **DBSMOA** offers both competitive accuracy and high robustness on the Page Blocks dataset.

D. 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 24 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.

TABLE 22: One-sample Page Blocks results for vehicle dataset

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

TABLE 23: 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

TABLE 24: 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 25: 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

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 25, 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.

To evaluate the computational efficiency of the optimization algorithms, a runtime analysis was conducted. Figure 41 presents a comparison of execution times for two algorithms, visually illustrating the mean and standard deviation of the time consumed during the experimental runs. The blue dashed line represents the time required by the first algorithm, while the red line corresponds to the second, with respective error bars indicating variability.

To further investigate the computational characteristics of the optimization methods, the distribution of execution times is examined. Figure 42 presents a histogram comparing the time intervals required by two algorithms across several runs. The histogram reveals variations in runtime frequency, providing insight into the computational stability and performance consistency of each method.

The performance of DBSMOA in terms of optimization fitness was also assessed under varying values of a and X. Table 26 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 27 further supports these findings. Both a and X demonstrated statistically significant deviations from the null hypothesis ($p < 0.0001$). The mean fitness values were 0.3617 and 0.3762, respectively, with

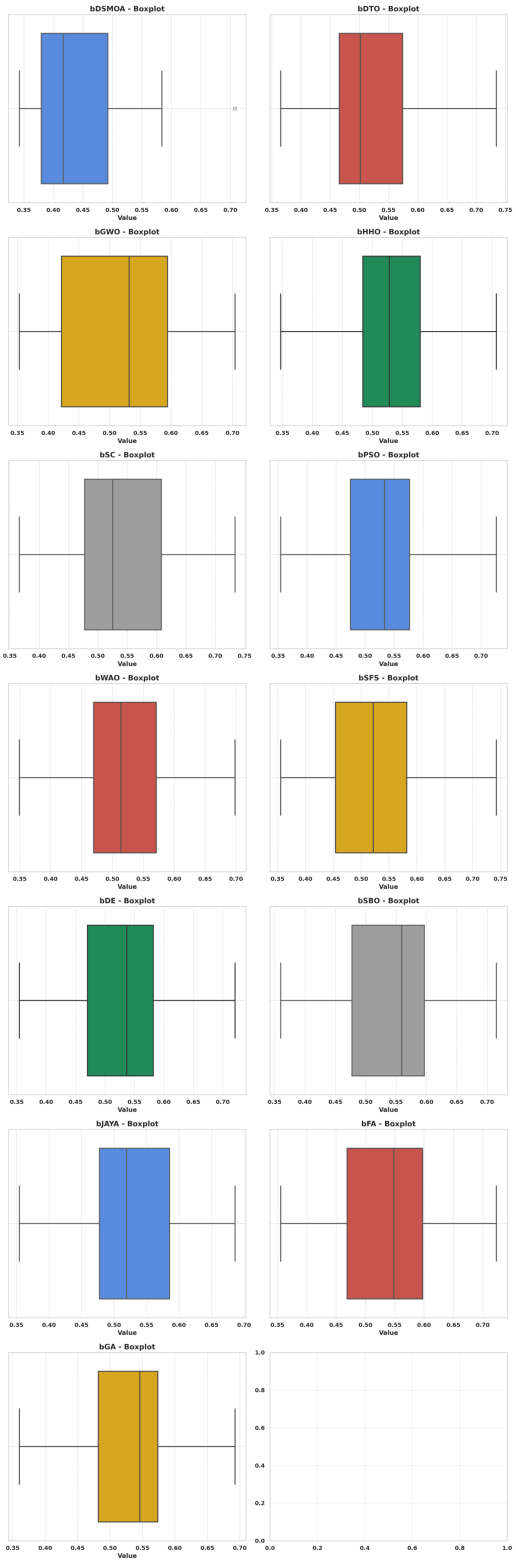

FIGURE 22: Grid of boxplots for individual feature selection algorithms. Each subplot displays the interquartile range, median, and whiskers for a specific method, allowing for a side-by-side evaluation of distribution characteristics.

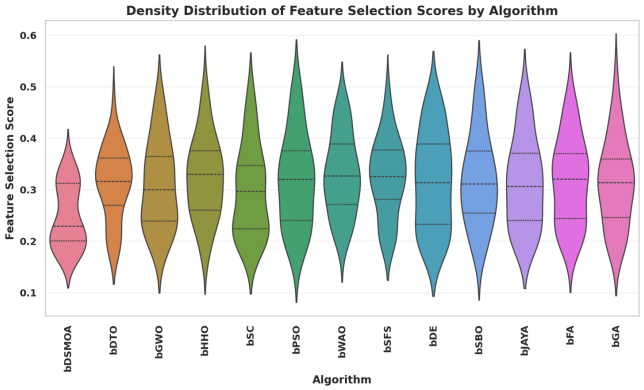

FIGURE 23: Violin plot displaying the density distribution of feature selection scores for each algorithm. Internal lines represent quartiles, while the width of each shape reflects the frequency of score values.

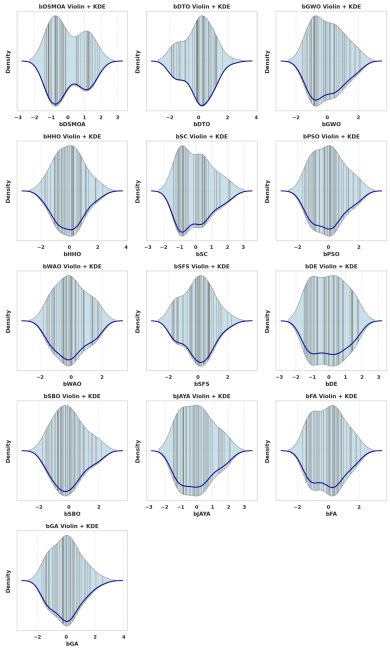

FIGURE 24: Grid of violin plots with KDE overlays for each algorithm. Each panel shows the density distribution along with individual score positions (rug plot), providing insights into score variability and concentration.

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.

This analysis confirms that parameter tuning is not merely a marginal concern but a central aspect of DBSMOA's effectiveness. From a runtime perspective, values around $a = 0.35$ and $X = 1.4$ minimize computational overhead, while the fitness-based analysis suggests that values near $a = 0.25$ and $X = 0.5$ yield the most accurate solutions. These results are statistically supported and can serve as a foundation for future adaptive or self-tuning strategies within the algorithm.

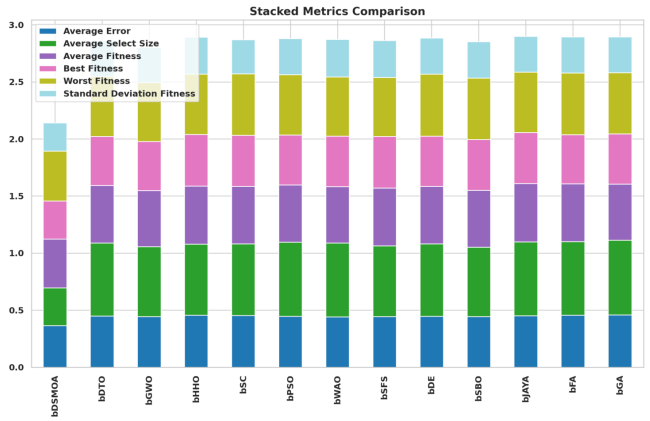

FIGURE 25: Stacked bar chart comparing six performance metrics across feature selection algorithms: Average Error, Average Select Size, Average Fitness, Best Fitness, Worst Fitness, and Standard Deviation of Fitness. This layout reveals overall profile differences between methods.

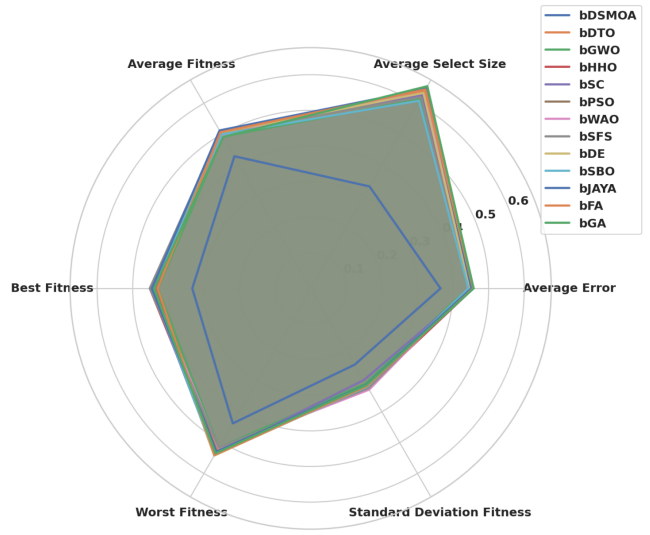

FIGURE 26: Radar plot comparing algorithms based on six normalized metrics: average error, average select size, average fitness, best fitness, worst fitness, and standard deviation of fitness. Each algorithm's performance is represented by a distinct polygon.

To compare the optimization performance in terms of solution quality, a statistical summary of the fitness values obtained by two different algorithms is presented. As shown in Figure 43, the plot illustrates the mean fitness value along with its standard deviation for each method. The use of error bars effectively conveys the variability and reliability of the results, offering a visual insight into the relative optimization strengths.

To evaluate the consistency and distribution of fitness values across different runs of the algorithms, Figure 44 displays a histogram of fitness values. This graphical summary highlights the frequency of fitness scores achieved by each

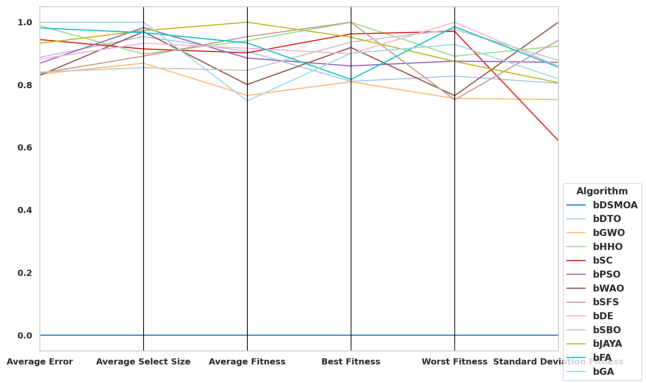

FIGURE 27: Parallel coordinates plot of normalized performance metrics for each algorithm. Each line represents an algorithm and spans axes for Average Error, Average Select Size, Average Fitness, Best Fitness, Worst Fitness, and Standard Deviation of Fitness.

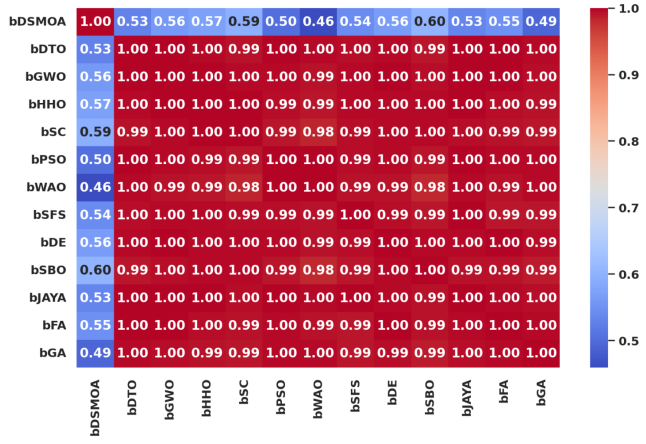

FIGURE 28: Correlation heatmap of normalized performance metrics between algorithms. Darker red cells indicate stronger positive correlations, while lighter or blue-toned cells indicate weaker alignment in metric profiles.

method, offering insight into their optimization behavior and performance spread.

To better understand the relationships between the selected value features across different clusters, Figure 45 presents a pair plot of the columns `a_Values` and `X_Values`, color-coded by cluster assignment. This visualization offers insight into the distribution and correlation of values within and between clusters, facilitating the interpretation of the clustering structure and feature variance.

To further differentiate the clusters based on their average characteristics, Figure 46 illustrates a radar plot showing the mean values of four key features—`a_Values`, `X_Values`, `a_Time`, and `X_Time`—for each identified cluster. This multi-dimensional visualization enables a quick comparative overview of feature prominence across clusters, aiding in the interpretation of cluster profiles.

In conclusion, sensitivity analysis has shown that careful

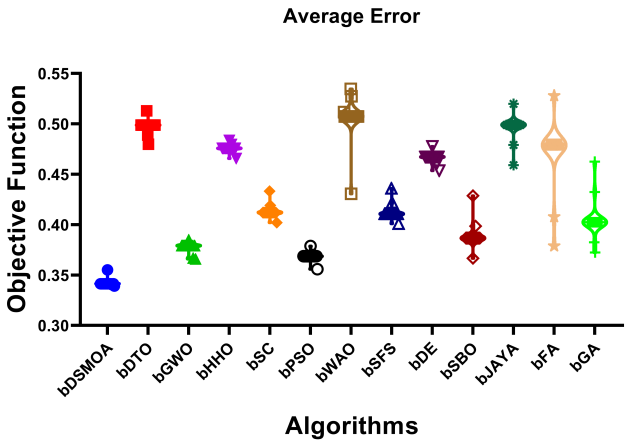

FIGURE 29: Violin plot showing the distribution of average error values for different algorithms applied to the WDBC dataset.

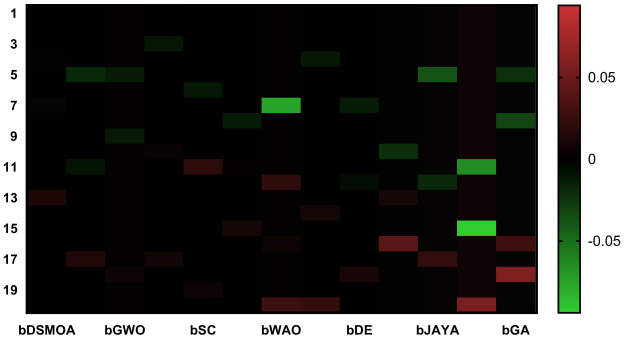

FIGURE 30: Heat map visualizing the outcome of ordinary one-way ANOVA tests across algorithms on the WDBC dataset. Each cell reflects the magnitude and direction of deviation for a given feature.

calibration of a and X can significantly enhance both the efficiency and effectiveness of DBSMOA.

E. 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 28 presents a detailed comparison of the average and standard deviation of convergence times, along with corresponding average error values across multiple benchmark datasets.

According to the results, DBSMOA exhibits a highly competitive profile in terms of both speed and reliability. On several datasets, it not only achieves some of the lowest average errors but also converges significantly faster than other methods. For instance, in the *Climate* dataset, DBSMOA achieves an average error of 0.31872, the lowest among all algorithms, with an exceptionally fast convergence time of just 0.063 seconds—outperforming all other techniques, many of which

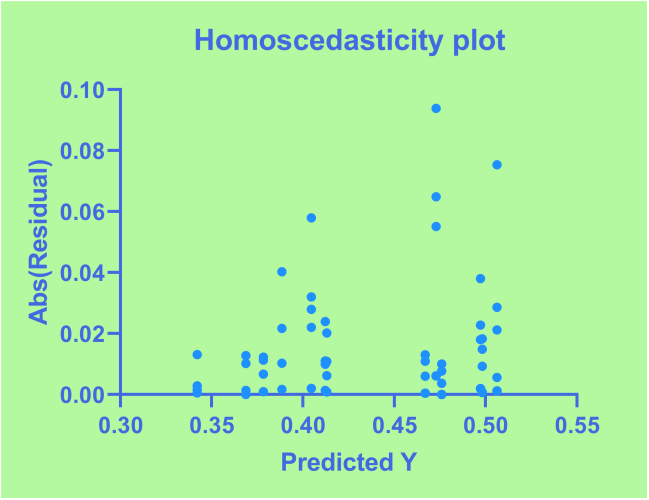

FIGURE 31: Homoscedasticity plot showing the absolute residuals versus predicted values for the WDBC dataset.

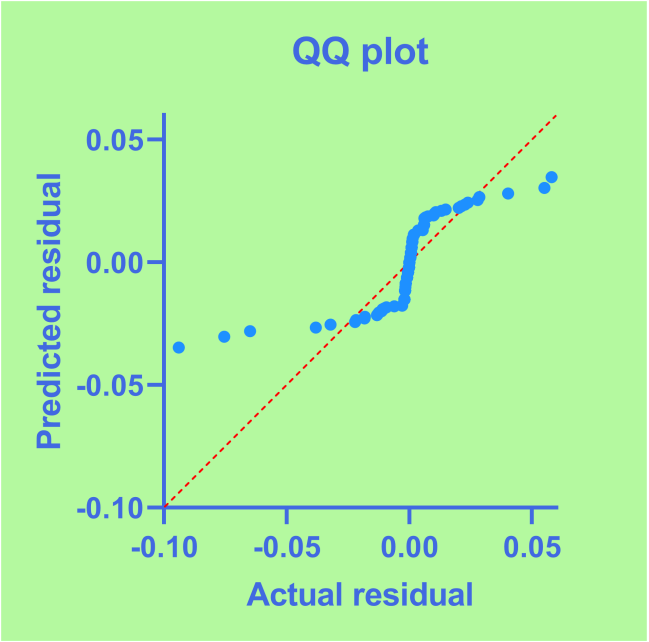

FIGURE 32: QQ plot for residuals of the WDBC dataset comparing predicted versus actual residuals to assess normality.

require over 2 seconds. This highlights DBSMOA's effectiveness and efficiency, especially in lightweight environments.

In more complex tasks, such as *Robot-failures-lp3*, DBSMOA achieves the best error rate (0.28038) while maintaining a low convergence time (0.993 seconds), compared to higher times and errors in alternatives like bGWO (2.328 seconds, 0.44628 error) or bPSO (2.098 seconds, 0.44758 error). Similarly, for the *Australian* dataset, DBSMOA yields the lowest average error (0.27898) with a shorter convergence time (1.792s) than all other methods, many of which exceed 3 seconds.

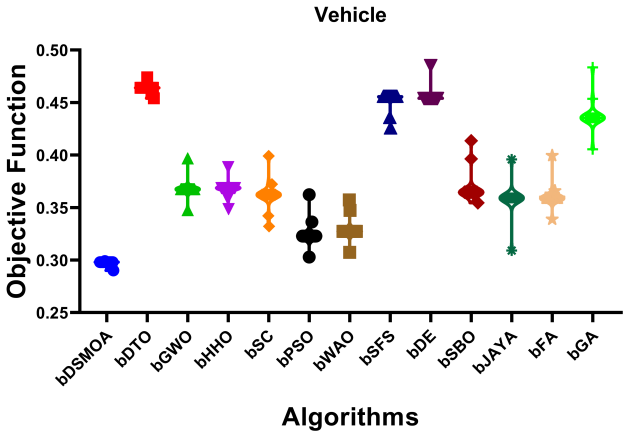

FIGURE 33: Objective function distribution across feature selection algorithms for the Vehicle dataset. Marker shapes and widths represent algorithm-specific performance spread and central values.

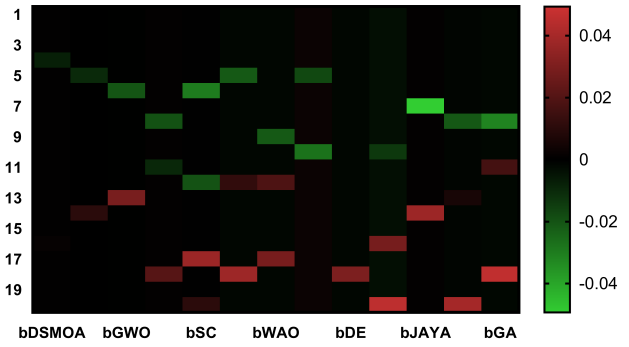

FIGURE 34: Heatmap showing deviation patterns from one-way ANOVA analysis on the Vehicle dataset. The color intensity represents the signed magnitude of deviation per algorithm and instance.

Even in high-dimensional or computationally intensive datasets, such as *Estimation of Obesity Levels*, DBSMOA maintains superior accuracy (0.29109) while converging faster (6.12s) than several alternatives, including bPSO (9.824s) and bGA (9.859s). This reflects DBSMOA's scalability and ability to balance computational cost with optimization quality. Beyond raw speed, DBSMOA also demonstrates stability, as reflected in its consistently low standard deviation in convergence time across multiple datasets. For example, on the *Breast Cancer Coimbra* dataset, it converges with a standard deviation of only 0.036 seconds—much lower than the fluctuations observed in algorithms like bSBO (0.129 seconds) or bDE (0.113 seconds). This reliability makes DBSMOA suitable for applications requiring predictable and repeatable performance.

To assess the computational efficiency of each optimizer across different benchmark functions, Figure 47 illustrates

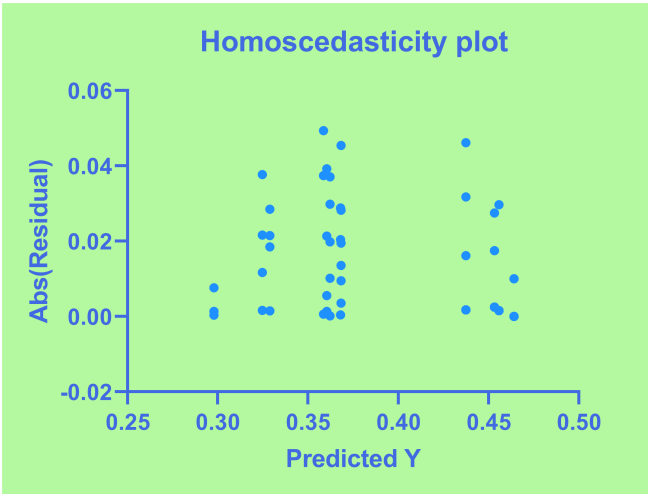

FIGURE 35: Homoscedasticity plot of residuals versus predicted values for the Vehicle dataset. The scatter pattern is examined for signs of non-constant variance.

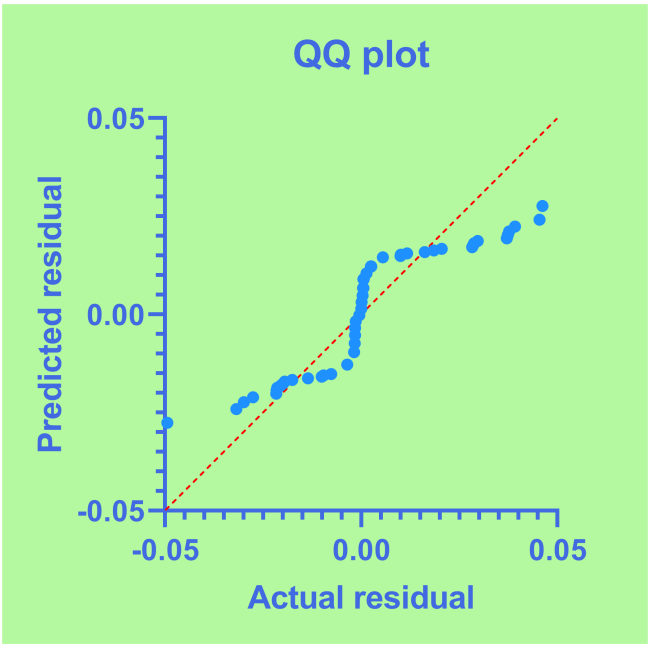

FIGURE 36: Q-Q plot showing the relationship between actual and predicted residuals for the Vehicle dataset. The red dashed line represents the theoretical normal quantile line.

the average time taken by each algorithm on a selection of objective functions. This time-based analysis helps identify performance bottlenecks and computational overheads associated with specific optimizers, offering insights into their scalability and runtime behavior.

To visualize the cumulative computational effort distributed across different benchmark functions and optimizers, Figure 48 presents a stacked area plot of the average time. This plot enables a comparative analysis of time consumption patterns. It highlights how individual functions contribute to

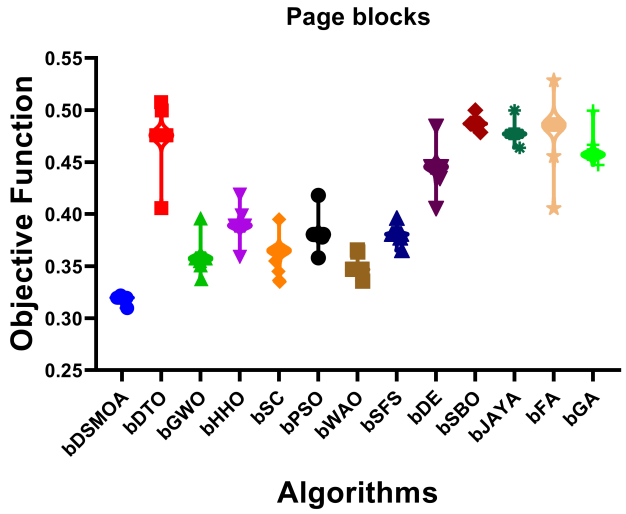

FIGURE 37: Distribution of objective function values for each algorithm on the Page Blocks dataset. Each point and shape reflects algorithm-specific performance variability and central tendency.

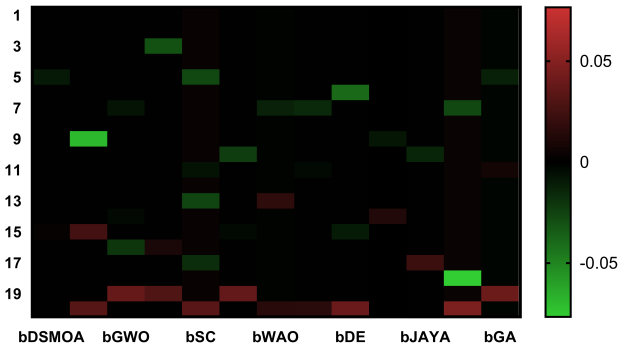

FIGURE 38: Heatmap of one-way ANOVA deviations for the Page Blocks dataset. Each cell displays the direction and magnitude of deviation for a given instance and algorithm, highlighting variability in performance across the dataset.

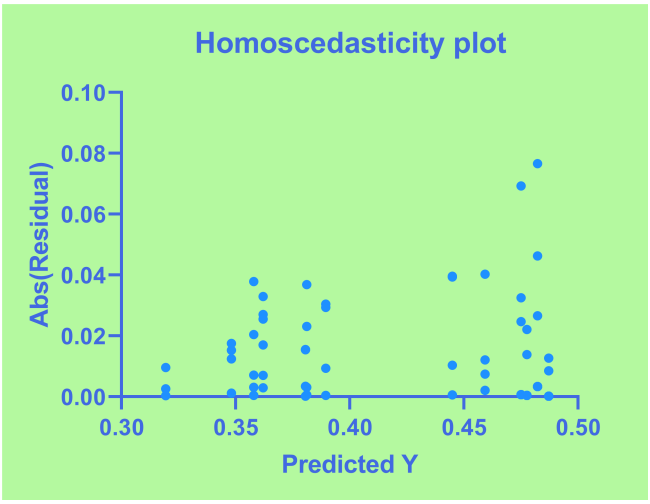

FIGURE 39: Homoscedasticity plot for the Page Blocks dataset under one-way ANOVA. The residuals are plotted against predicted values to assess constant variance across predictions.

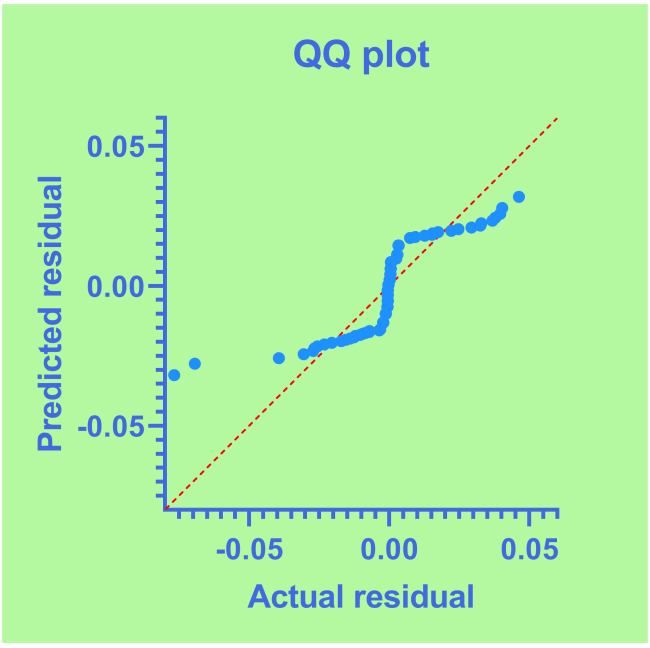

FIGURE 40: Q-Q plot comparing actual residuals against expected normal quantiles for the Page Blocks dataset. The red line represents the ideal 1:1 fit expected under normal distribution.

the overall runtime per optimizer, providing valuable insights into performance scalability and load distribution characteristics.

To further understand the distribution of computational time across various benchmark functions, Figure 49 illustrates a violin plot for average time. This visualization combines a box plot with a kernel density estimation, allowing a clear inspection of both the central tendency and the spread of time consumption for each function. It highlights the functions that exhibit higher computational costs and those with broader or skewed distributions.

To assess the distributional characteristics of average execution times across various benchmark functions, Figure 50 presents a kernel density estimation (KDE) plot. This KDE plot visualizes the distribution of average time values for each function, enabling the identification of peaks, skewness, and the spread of time performance. The curves for each function provide comparative insight into computational intensity and variability, highlighting which functions are consistently efficient and which ones pose higher temporal costs.

To further explore the computational efficiency of different algorithms across the benchmark functions, Figure 51 presents a radar chart of the average execution time. This vi-

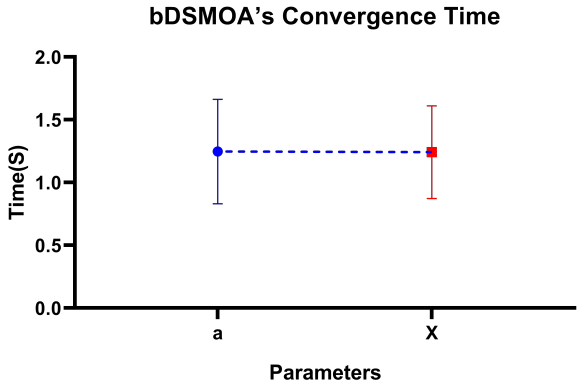

FIGURE 41: Comparison of average execution time (with standard deviation) between two optimization algorithms.

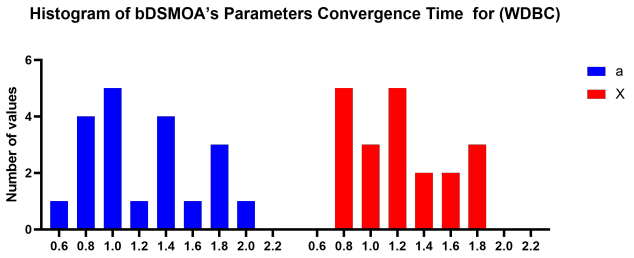

FIGURE 42: Histogram showing the distribution of execution times for two different algorithms.

TABLE 26: 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

sualization enables a comparative overview of time consumption patterns by displaying how each algorithm performs across all functions (D1–D48). Peaks in the radar spokes highlight functions that are more computationally intensive for specific algorithms, while compact profiles indicate uniformly efficient behavior. The chart supports simultaneous

TABLE 27: 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

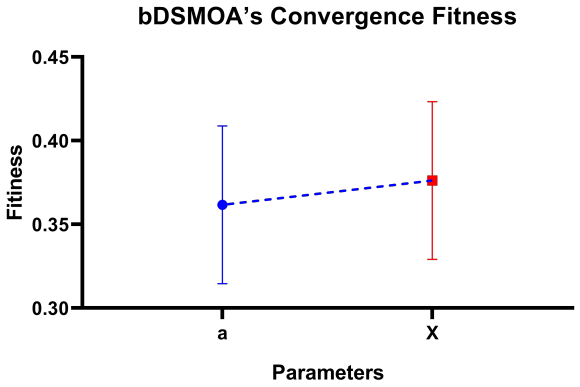

FIGURE 43: Comparison of average fitness values with standard deviation for two algorithms.

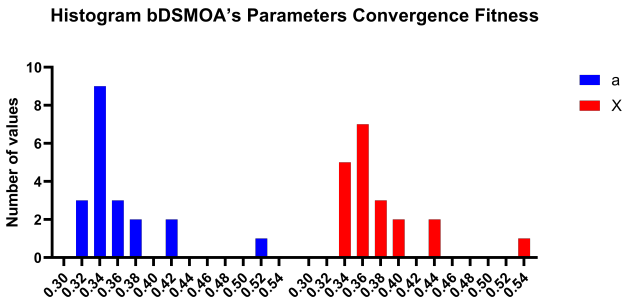

FIGURE 44: Histogram illustrating the distribution of fitness values for two optimization algorithms across multiple runs.

multidimensional comparison in a single frame, making it a valuable diagnostic tool for time-based performance assessment.

To gain a detailed view of the distribution of average execution times across benchmark functions, Figure 52 illustrates a swarm plot. This visualization enables the examination of the individual time performance of each algorithm on every function, displaying the spread and clustering of execution times. The use of non-overlapping points highlights

TABLE 28: 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

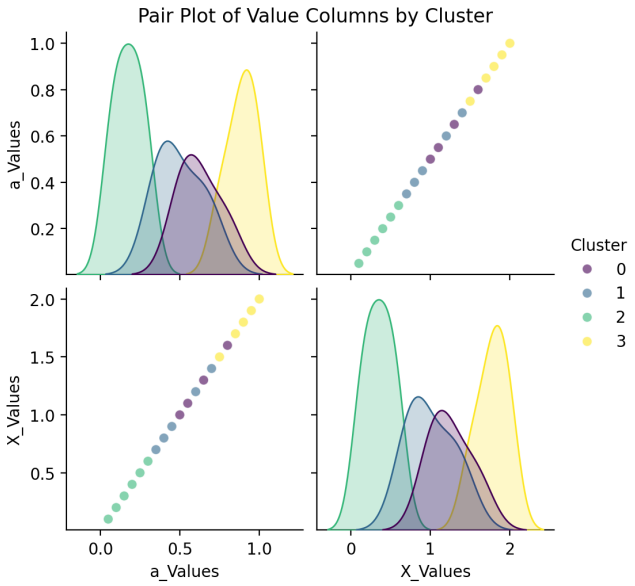

FIGURE 45: Pair plot showing the distribution and relationship between `a_Values` and `X_Values` grouped by cluster.

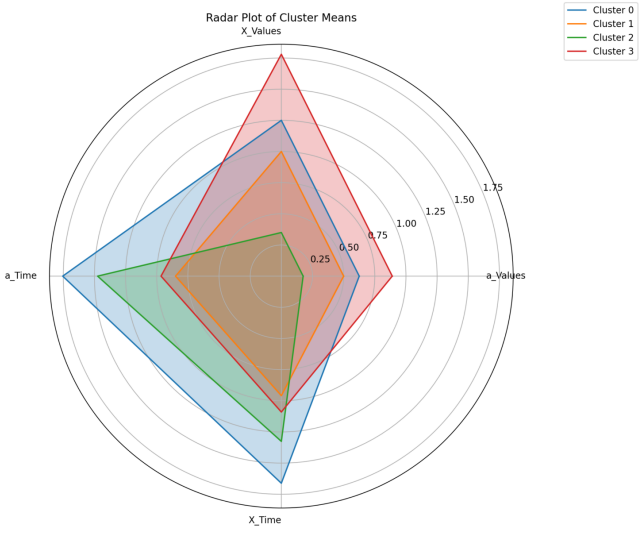

FIGURE 46: Radar plot of mean feature values for each cluster, highlighting inter-cluster variability across the dimensions `a_Values`, `X_Values`, `a_Time`, and `X_Time`.

local variations and density, facilitating the identification of patterns such as consistent performance, outliers, or bottle-

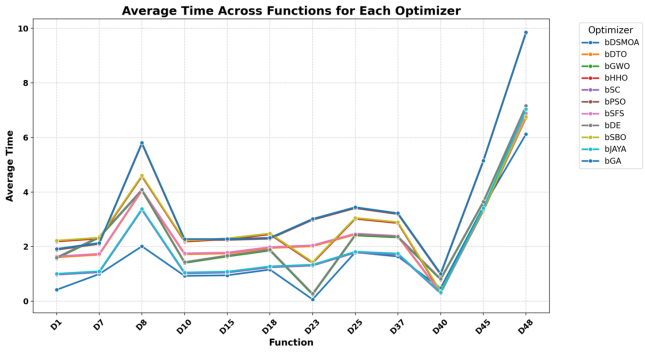

FIGURE 47: Average computation time for each optimizer across multiple benchmark functions.

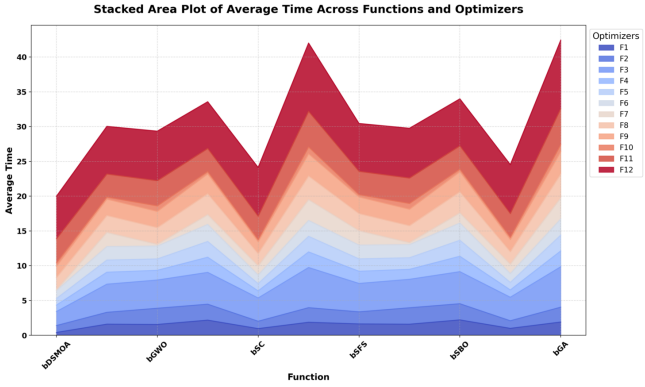

FIGURE 48: Stacked area plot showing the average computation time per function across selected optimizers.

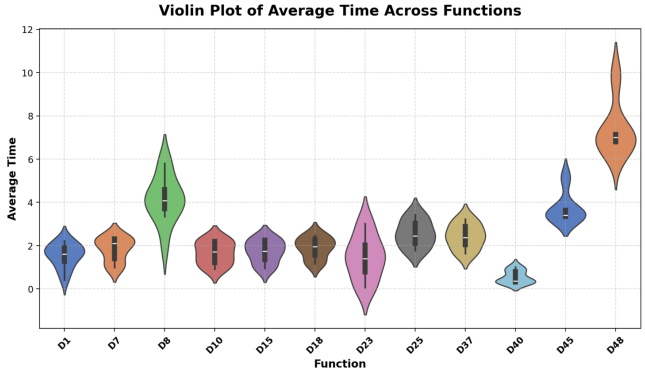

FIGURE 49: Violin plot showing the distribution of average computational time across different functions.

necks across the test suite.

To better understand how each algorithm performs over a subset of benchmark functions, Figure 53 presents a series of stacked area plots. Each subfigure corresponds to a specific optimizer and shows the average execution time across selected functions. These plots enable a side-by-side comparison of trends within individual algorithms, making it easier to detect which functions are computationally demanding for each method. The visualization helps identify whether

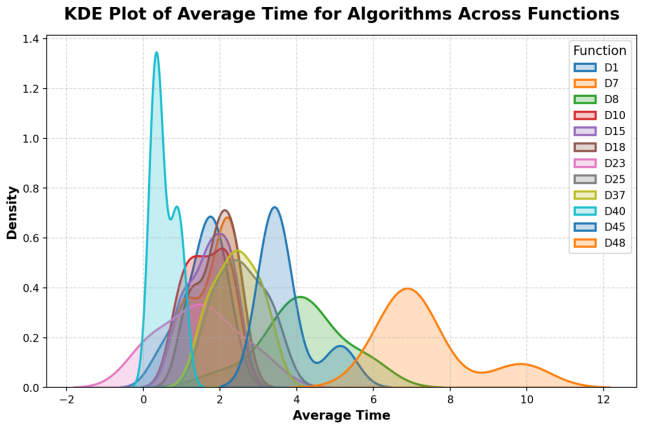

FIGURE 50: KDE plot of the average time for algorithms across benchmark functions.

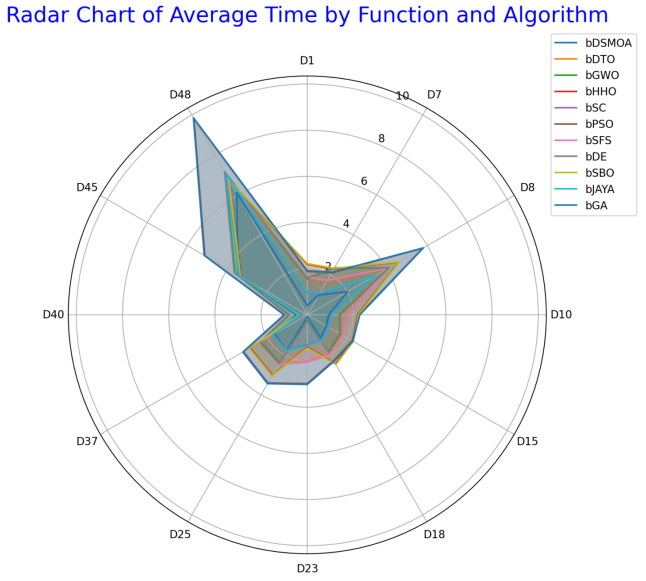

FIGURE 51: Radar chart of average time across functions and algorithms.

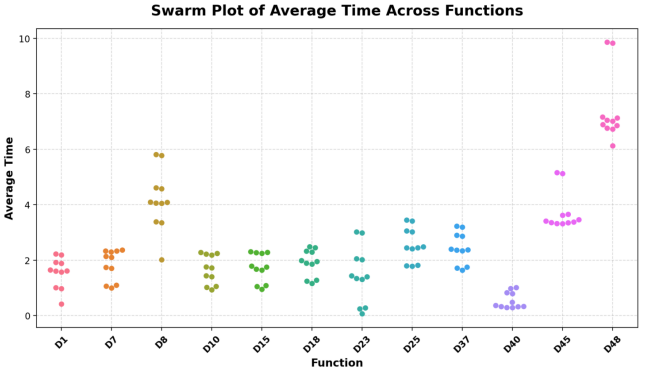

FIGURE 52: Swarm plot of average time across functions.

performance inconsistencies are due to algorithmic design or

task-specific complexity.

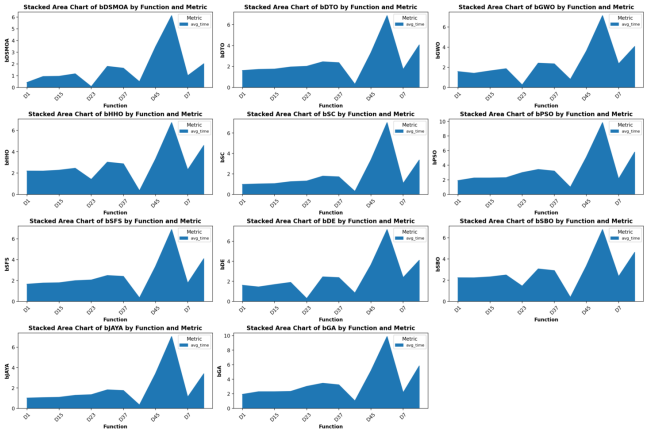

FIGURE 53: Stacked area plots of average time per function for each optimizer.

Overall, the results in Table 28 strongly indicate that DBSMOA offers an excellent balance of fast convergence and high accuracy. These attributes are particularly valuable in real-world scenarios where computational time is limited and precision is crucial.

V. DISCUSSION

The empirical results obtained from applying the proposed Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA) demonstrate several significant advantages in the context of high-dimensional feature selection problems. These advantages can be attributed to the algorithm's dynamic behavior, binary encoding, and biologically inspired strategy, all of which contributed to its strong performance across multiple datasets and evaluation metrics.

One of the most prominent aspects of DBSMOA is its adaptability, which arises from the dynamic reassignment of exploration and exploitation roles based on population diversity and convergence trends. This mechanism prevented premature convergence, a common drawback in traditional binary metaheuristics, such as Binary Particle Swarm Optimization (bPSO) and Binary Genetic Algorithm (bGA). By avoiding static control parameters and incorporating feedback-driven dynamics, DBSMOA maintained a balance between global exploration and local refinement throughout the search process, leading to higher-quality solutions in terms of classification accuracy and feature reduction.

Another key contributor to DBSMOA's performance is the elitism strategy, which preserved historically optimal solutions and reintroduced them when beneficial. This approach improved solution stability across independent runs and minimized the risk of fitness degradation due to random fluctuations. Furthermore, the sigmoid-based transfer function effectively bridged the continuous dynamics of the original SMOA with the binary nature of feature selection tasks. This probabilistic mapping provided smooth control over binary transitions, enabling DBSMOA to retain the underlying advantages of movement-driven swarm intelligence within a discrete search space.

In addition to the accuracy and compactness of the selected feature subsets, DBSMOA achieved competitive computational efficiency and resource usage. The algorithm's runtime and memory profile remained moderate, which is especially important in real-world applications involving large-scale or resource-constrained environments. Compared to other algorithms, DBSMOA consistently ranks among the top performers across composite evaluation metrics, including average time, standard deviation, CPU usage, and efficiency score.

The generalization capability of DBSMOA was also evident from its robust performance across diverse datasets, which varied in dimensionality, class distributions, and domain characteristics. Its ability to consistently yield stable and compact feature subsets with minimal performance loss suggests that DBSMOA is not only effective in benchmark settings but also adaptable to real-world scenarios involving noisy, redundant, or imbalanced data.

However, while DBSMOA outperformed many state-of-the-art algorithms, the discussion also invites consideration of possible enhancements. For instance, the use of alternative transfer functions or hybridization with local search strategies could further improve its convergence speed or accuracy. Moreover, the current framework uses a static fitness weight to balance accuracy and subset size; an adaptive or multi-objective formulation may offer greater flexibility in scenarios with dynamic performance constraints or multi-criteria decision-making requirements.

In summary, DBSMOA's dynamic nature, binary encoding, and biologically inspired operations contributed significantly to its superior performance in feature selection tasks. The discussion reinforces the algorithm's potential as a scalable, efficient, and accurate optimizer for high-dimensional binary search problems.

VI. CONCLUSION AND FUTURE DIRECTIONS

This paper proposed a novel nature-inspired optimization method, the Dynamic Binary Swordfish Movement Optimization Algorithm (DBSMOA), for efficient and scalable feature selection in high-dimensional binary search spaces. Building upon the original Swordfish Movement Optimization Algorithm (SMOA), the proposed approach introduced several key innovations: dynamic behavioral adaptation through population role transitions, probabilistic binary mapping via a sigmoid-based transfer function, and elitist selection mechanisms to retain high-quality solutions. These enhancements enabled DBSMOA to address common limitations in existing binary metaheuristics, such as premature convergence, static control parameters, and sensitivity to local optima.

Comprehensive experimental evaluations on benchmark datasets demonstrated that DBSMOA consistently delivered competitive or superior performance in terms of classification accuracy, feature subset compactness, computational efficiency, and robustness across multiple runs. Its ability to

dynamically adjust its search behavior based on feedback from the population and fitness landscape contributed to its high adaptability and generalization across diverse data scenarios.

While DBSMOA offers promising results, several avenues for future research remain open. First, the integration of alternative transfer functions or adaptive transfer mechanisms may enhance its binary decision dynamics. Second, hybridizing DBSMOA with local search strategies could further refine convergence in the exploitation phase. Third, extending the current single-objective formulation to a multi-objective framework would enable simultaneous optimization of conflicting goals, such as maximizing accuracy while minimizing computational cost. Finally, applying DBSMOA to other binary optimization problems beyond feature selection—such as subset selection, binary classification rule mining, or combinatorial design—could further validate its versatility and effectiveness.

In conclusion, DBSMOA represents a significant advancement in dynamic binary metaheuristic optimization, offering a biologically grounded, flexible, and high-performing solution for complex feature selection tasks in real-world data environments.

VII. ACKNOWLEDGMENT

Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R308), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia

REFERENCES

[1] S. Weiss and N. Indurkhya, "Predictive data mining: A practical guidemorgan," 1998.
[2] C. Ding and H. Peng, "Minimum redundancy feature selection from microarray gene expression data," Journal of bioinformatics and computational biology, vol. 3, no. 02, pp. 185–205, 2005.
[3] Z. Gao, S. X. Ding, and C. Cecati, "Real-time fault diagnosis and fault-tolerant control," IEEE Transactions on industrial Electronics, vol. 62, no. 6, pp. 3752–3756, 2015.
[4] B. Demir and S. Erturk, "Hyperspectral image classification using relevance vector machines," IEEE Geoscience and Remote Sensing Letters, vol. 4, no. 4, pp. 586–590, 2007.
[5] B. Demir and S. Erturk, "Hyperspectral image classification using relevance vector machines," IEEE Geoscience and Remote Sensing Letters, vol. 4, no. 4, pp. 586–590, 2007.
[6] H. Liu and H. Motoda, Feature selection for knowledge discovery and data mining, vol. 454. Springer science & business media, 2012.
[7] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," Journal of machine learning research, vol. 3, no. Mar, pp. 1157–1182, 2003.
[8] L. Yu and H. Liu, "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] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," ACM computing surveys (CSUR), vol. 50, no. 6, pp. 1–45, 2017.
[10] M. Dash and H. Liu, "Feature selection for classification," Intelligent data analysis, vol. 1, no. 1-4, pp. 131–156, 1997.
[11] P. Pudil, J. Novovičová, and J. Kittler, "Floating search methods in feature selection," Pattern recognition letters, vol. 15, no. 11, pp. 1119–1125, 1994.
[12] A. Jain and D. Zongker, "Feature selection: Evaluation, application, and small sample performance," IEEE transactions on pattern analysis and machine intelligence, vol. 19, no. 2, pp. 153–158, 2002.
[13] D. E. Goldberg, "Optimization, and machine learning," Genetic algorithms in Search, 1989.
[14] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Proceedings of ICNN'95-international conference on neural networks, vol. 4, pp. 1942–1948, ieee, 1995.
[15] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by simulated annealing," science, vol. 220, no. 4598, pp. 671–680, 1983.
[16] X.-S. Yang, Nature-inspired metaheuristic algorithms. Luniver press, 2010.
[17] M. Dorigo and T. Stutzle, "Ant colony optimization. mit press, cambridge, ma," MIT Press, Cambridge, MA., 2004.
[18] F. W. Glover and G. A. Kochenberger, Handbook of metaheuristics, vol. 57. Springer Science & Business Media, 2003.
[19] L. Davis, "Handbook of genetic algorithms, van nostrand reinhold," 1994.
[20] S. Mirjalili, "How effective is the grey wolf optimizer in training multi-layer perceptrons," Applied intelligence, vol. 43, pp. 150–161, 2015.
[21] H. Faris, A.-Z. Ala'M, A. A. Heidari, I. Aljarah, M. Mafarja, M. A. Hassonah, and H. Fujita, "An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks," Information Fusion, vol. 48, pp. 67–83, 2019.
[22] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," IEEE transactions on evolutionary computation, vol. 10, no. 6, pp. 646–657, 2006.
[23] H. K. Hamarashid, B. A. Hassan, and T. A. Rashid, "Modified-improved fitness dependent optimizer for complex and engineering problems," Knowledge-based systems, vol. 300, p. 112098, 2024.
[24] R. Poli, "Analysis of the publications on the applications of particle swarm optimisation," Journal of Artificial Evolution and Applications, vol. 2008, no. 1, p. 685175, 2008.
[25] E. A. K. Zaman, A. Ahmad, and A. Mohamed, "Adaptive threshold optimisation for online feature selection using dynamic particle swarm optimisation in determining feature relevancy and redundancy," Applied Soft Computing, vol. 156, p. 111477, 2024.
[26] B. Madhusudhanan, P. Sumathi, N. S. Karpagam, A. Mahesh, and P. A. P. Suhi, "An hybrid metaheuristic approach for efficient feature selection," Cluster Computing, vol. 22, no. Suppl 6, pp. 14541–14549, 2019.
[27] M. A. Hall, Correlation-based feature selection for machine learning. PhD thesis, The University of Waikato, 1999.
[28] S. S. Kareem, R. R. Mostafa, F. A. Hashim, and H. M. El-Bakry, "An effective feature selection model using hybrid metaheuristic algorithms for iot intrusion detection," Sensors, vol. 22, no. 4, p. 1396, 2022.
[29] R. Sugumar, "Rough set theory-based feature selection and fga-nn classifier for medical data classification," 2019.
[30] A. K. Jain and B. Chandrasekaran, "39 dimensionality and sample size considerations in pattern recognition practice," Handbook of statistics, vol. 2, pp. 835–855, 1982.
[31] A. Liaw, M. Wiener, et al., "Classification and regression by randomforest," R news, vol. 2, no. 3, pp. 18–22, 2002.
[32] W. Li, Y. Chai, F. Khan, S. R. U. Jan, S. Verma, V. G. Menon, f. Kavita, and X. Li, "A comprehensive survey on machine learning-based big data analytics for iot-enabled smart healthcare system," Mobile networks and applications, vol. 26, pp. 234–252, 2021.
[33] H. Lategahn, A. Geiger, and B. Kitt, "Visual slam for autonomous ground vehicles," in 2011 IEEE International Conference on Robotics and Automation, pp. 1732–1737, IEEE, 2011.
[34] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.
[35] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in International conference on machine learning, pp. 6105–6114, PMLR, 2019.
[36] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," IEEE Transactions on knowledge and data engineering, vol. 17, no. 4, pp. 491–502, 2005.
[37] Y. Saeys, I. Inza, and P. Larranaga, "A review of feature selection techniques in bioinformatics," bioinformatics, vol. 23, no. 19, pp. 2507–2517, 2007.
[38] A. Tawhid, T. Teotia, and H. Elmiligi, "Machine learning for optimizing healthcare resources," in Machine Learning, Big Data, and IoT for Medical Informatics, pp. 215–239, Elsevier, 2021.
[39] N. Abd-Alsabour and S. Ramakrishnan, "Hybrid metaheuristics for classification problems," Pattern Recognition-Analysis and Applications, vol. 10, p. 65253, 2016.

[40] M. Sharma and P. Kaur, "A comprehensive analysis of nature-inspired meta-heuristic techniques for feature selection problem," Archives of Computational Methods in Engineering, vol. 28, pp. 1103–1127, 2021.

[41] H. Du, Z. Wang, W. Zhan, and J. Guo, "Elitism and distance strategy for selection of evolutionary algorithms," IEEe Access, vol. 6, pp. 44531–44541, 2018.

[42] A. Mukhopadhyay, U. Maulik, S. Bandyopadhyay, and C. A. C. Coello, "A survey of multiobjective evolutionary algorithms for data mining: Part i," IEEE Transactions on Evolutionary Computation, vol. 18, no. 1, pp. 4–19, 2013.

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

[44] J. Hua, W. D. Tembe, and E. R. Dougherty, "Performance of feature-selection methods in the classification of high-dimension data," Pattern Recognition, vol. 42, no. 3, pp. 409–424, 2009.

[45] G. Wei, J. Zhao, Y. Feng, A. He, and J. Yu, "A novel hybrid feature selection method based on dynamic feature importance," Applied Soft Computing, vol. 93, p. 106337, 2020.

[46] H. Salimi, "Stochastic fractal search: A powerful metaheuristic algorithm," Knowledge-Based Systems, vol. 75, pp. 1–18, 2015.

[47] C. R. Reeves, "Genetic algorithms," in Handbook of Metaheuristics (M. Gendreau and J.-Y. Potvin, eds.), pp. 109–139, Springer US, 2010.

[48] A. Takieldeen, E.-S. El-kenawy, M. Hadwan, and R. Zaki, "Dipper throated optimization algorithm for unconstrained function and feature selection," Computers, Materials & Continua, vol. 72, no. 1, pp. 1465–1481, 2022.

[49] M. A. Awadallah, M. A. Al-Betar, A. I. Hammouri, and O. A. Alomari, "Binary jaya algorithm with adaptive mutation for feature selection," Arabian Journal for Science and Engineering, vol. 45, no. 12, pp. 10875–10890, 2020.

[50] I. Kucukoglu, "Binary satin bowerbird optimizer for the set covering problem," in Industrial Engineering in the Digital Disruption Era (F. Calisir and O. Korhan, eds.), pp. 73–86, Springer International Publishing, 2020.

[51] T. Thaher, A. A. Heidari, M. Mafarja, J. S. Dong, and S. Mirjalili, "Binary harris hawks optimizer for high-dimensional, low sample size feature selection," in Evolutionary Machine Learning Techniques: Algorithms and Applications (S. Mirjalili, H. Faris, and I. Aljarah, eds.), pp. 251–272, Springer, 2020.

[52] S. M. Azzam, O. E. Emam, and A. S. Abolaban, "An improved differential evolution with sailfish optimizer (desfo) for handling feature selection problem," Scientific Reports, vol. 14, no. 1, p. 13517, 2024.

[53] J. L. J. Pereira, M. B. Francisco, B. J. Ma, G. F. Gomes, and A. C. Lorena, "Golden lichtenberg algorithm: a fibonacci sequence approach applied to feature selection," Neural Computing and Applications, vol. 36, no. 32, pp. 20493–20511, 2024.

[54] W. Xie, L. Wang, K. Yu, T. Shi, and W. Li, "Improved multi-layer binary firefly algorithm for optimizing feature selection and classification of microarray data," Biomedical Signal Processing and Control, vol. 79, p. 104080, 2023.

[55] G. Pampara, A. P. Engelbrecht, and N. Franken, "Binary differential evolution," in 2006 IEEE International Conference on Evolutionary Computation, pp. 1873–1879, 2006.

[56] A. G. Hussien, D. Oliva, E. H. Houssein, A. A. Juan, and X. Yu, "Binary whale optimization algorithm for dimensionality reduction," Mathematics, vol. 8, no. 10, p. 1821, 2020.

[57] B. H. Nguyen, B. Xue, P. Andreae, and M. Zhang, "A new binary particle swarm optimization approach: Momentum and dynamic balance between exploration and exploitation," IEEE Transactions on Cybernetics, vol. 51, no. 2, pp. 589–603, 2021.

[58] K. S. Reddy, L. K. Panwar, B. Panigrahi, and R. Kumar, "A new binary variant of sine–cosine algorithm: Development and application to solve profit-based unit commitment problem," Arabian Journal for Science and Engineering, vol. 43, no. 8, pp. 4041–4056, 2018.

[59] L. Kumar and K. K. Bharti, "A novel hybrid bpso–sca approach for feature selection," Natural Computing, vol. 20, no. 1, pp. 39–61, 2021.

[60] M. Alweshah, S. A. Khalaileh, B. B. Gupta, A. Almomani, A. I. Hammouri, and M. A. Al-Betar, "The monarch butterfly optimization algorithm for solving feature selection problems," Neural Computing and Applications, vol. 34, no. 14, pp. 11267–11281, 2022.

[61] A. A. A. El-Mageed, A. A. Abohany, and K. M. Hosny, "Enhanced binary kepler optimization algorithm for effective feature selection of supervised learning classification," Journal of Big Data, vol. 12, no. 1, p. 93, 2025.

[62] R. Ranjan and J. K. Chhabra, "Automatic feature selection using enhanced dynamic crow search algorithm," International Journal of Information Technology, vol. 15, no. 5, pp. 2777–2782, 2023.

[63] N. Khodadadi, E. Khodadadi, Q. Al-Tashi, E.-S. M. El-Kenawy, L. Abualigah, S. J. Abdulkadir, A. Alqushaibi, and S. Mirjalili, "Baoa: Binary arithmetic optimization algorithm with k-nearest neighbor classifier for feature selection," IEEE Access, vol. 11, pp. 94094–94115, 2023.

[64] E. Pashaei and E. Pashaei, "An efficient binary chimp optimization algorithm for feature selection in biomedical data classification," Neural Computing and Applications, vol. 34, no. 8, pp. 6427–6451, 2022.

[65] M. Li, Q. Luo, and Y. Zhou, "Bgoa-tvg: Binary grasshopper optimization algorithm with time-varying gaussian transfer functions for feature selection," Biomimetics, vol. 9, no. 3, 2024.

[66] N. H. Shikoun, A. S. Al-Eraqi, and I. S. Fathi, "Bincoa: An efficient binary crayfish optimization algorithm for feature selection," IEEE Access, vol. 12, pp. 28621–28635, 2024.

[67] G. Liu, Z. Guo, W. Liu, F. Jiang, and E. Fu, "A feature selection method based on the golden jackal-grey wolf hybrid optimization algorithm," PLOS ONE, vol. 19, pp. 1–32, 01 2024.

[68] M. Tubishat, M. Alswaitti, S. Mirjalili, M. A. Al-Garadi, M. T. Alrashdan, and T. A. Rana, "Dynamic butterfly optimization algorithm for feature selection," IEEE Access, vol. 8, pp. 194303–194314, 2020.

[69] A. G. Gad, K. M. Sallam, R. K. Chakrabortty, M. J. Ryan, and A. A. Abohany, "An improved binary sparrow search algorithm for feature selection in data classification," Neural Computing and Applications, vol. 34, no. 18, pp. 15705–15752, 2022.

[70] O. Akinola, O. N. Oyelade, and A. E. Ezugwu, "Binary ebola optimization search algorithm for feature selection and classification problems," Applied Sciences, vol. 12, no. 22, p. 11787, 2022.

[71] L. Abualigah and A. Diabat, "Chaotic binary group search optimizer for feature selection," Expert Systems with Applications, vol. 192, p. 116368, 2022.

[72] A. A. Abd El-Mageed, A. A. Abohany, and A. Elashry, "Effective feature selection strategy for supervised classification based on an improved binary aquila optimization algorithm," Computers & Industrial Engineering, vol. 181, p. 109300, 2023.

[73] T. Thaher, H. Chantar, J. Too, M. Mafarja, H. Turabieh, and E. H. Houssein, "Boolean particle swarm optimization with various evolutionary population dynamics approaches for feature selection problems," Expert Systems with Applications, vol. 195, p. 116550, 2022.

[74] S. S. M. Ghoneim, T. A. Farrag, A. A. Rashed, E.-S. M. El-Kenawy, and A. Ibrahim, "Adaptive dynamic meta-heuristics for feature selection and classification in diagnostic accuracy of transformer faults," IEEE Access, vol. 9, pp. 78324–78340, 2021.

[75] A. C. Cinar, "A novel adaptive memetic binary optimization algorithm for feature selection," Artificial Intelligence Review, vol. 56, no. 11, pp. 13463–13520, 2023.

[76] M. El-Sayed, A. Ali, D. Sami, H. Amal, A. Sarah, A. Abdelaziz, I. Abdelhameed, and M. Marwa, "A novel binary swordfish movement optimization algorithm (bsmoa) for efficient feature selection," Fusion: Practice and Applications, pp. 170–186, 2025.

[77] A. E. Takieldeen, E.-S. M. El-kenawy, M. Hadwan, and R. M. Zaki, "Dipper throated optimization algorithm for unconstrained function and feature selection," Comput. Mater. Contin, vol. 72, pp. 1465–1481, 2022.

[78] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," Advances in engineering software, vol. 69, pp. 46–61, 2014.

[79] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," Future generation computer systems, vol. 97, pp. 849–872, 2019.

[80] S. Mirjalili, "Sca: a sine cosine algorithm for solving optimization problems," Knowledge-based systems, vol. 96, pp. 120–133, 2016.

[81] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Proceedings of ICNN'95-international conference on neural networks, vol. 4, pp. 1942–1948, ieee, 1995.

[82] S. Mirjalili and A. Lewis, "The whale optimization algorithm," Advances in engineering software, vol. 95, pp. 51–67, 2016.

[83] H. Salimi, "Stochastic fractal search: a powerful metaheuristic algorithm," Knowledge-based systems, vol. 75, pp. 1–18, 2015.

[84] V. Feoktistov, Differential evolution. Springer, 2006.

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

[86] R. V. Rao, "Jaya: an advanced optimization algorithm and its engineering applications," 2019.
[87] N. F. Johari, A. M. Zain, M. H. Noorfa, and A. Udin, "Firefly algorithm for optimization problem," Applied Mechanics and Materials, vol. 421, pp. 512–517, 2013.
[88] S. Sivanandam, S. Deepa, S. Sivanandam, and S. Deepa, Genetic algorithms. Springer, 2008.
[89] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: A review," IEEE Transactions on pattern analysis and machine intelligence, vol. 22, no. 1, pp. 4–37, 2000.
[90] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," Machine learning, vol. 46, pp. 389–422, 2002.

FARIS H. RIZK is a researcher with over 2.5 years of experience as a Research Assistant at the Delta Higher Institute for Engineering & Technology (DHIET). He has worked under the supervision of Prof. El-Sayed M. El-Kenawy and Prof. Marwa M. Eid at DHIET, as well as Dr. Nima Khodadadi from the University of California, Berkeley. His research interests include metaheuristic optimization algorithms, computer vision, and applied machine learning. He is currently pursuing a B.Sc. in Communications and Electronics Engineering at DHIET, with an expected graduation in 2026.

KHALED SH. GABER is a research assistant at the Computer Science and Intelligent Systems Research Center in Virginia. His research interests include deep learning and metaheuristic optimization.

MARWA M. EID (Senior Member, IEEE) received the Ph.D. degree in electronics and communications engineering from the Faculty of Engineering, Mansoura University, Egypt, in 2015. From 2011 to 2021, she served as an Assistant Professor at the Delta Higher Institute for Engineering and Technology. Since 2022, she has been with the Faculty of Artificial Intelligence at Delta University for Science and Technology, Mansoura, Egypt, where she holds the position of Assistant Professor. Her research interests include image processing, encryption, wireless communication systems, and applications of field-programmable gate arrays (FPGAs).

DOAA SAMI KHAFAGA received the B.Sc. (Hons.) degree in Computer and Information Sciences and in Computer Science, and the M.Sc. and Ph.D. degrees in Computer Science from the Faculty of Computers and Artificial Intelligence, Helwan University, Egypt, in 2003, 2008, and 2013, respectively. She has over 18 years of academic experience and has served in various institutions, including the Science Department at the College of Information Technology and Artificial Intelligence, Misr University, Egypt; the Computer Science Department at the Institute of Public Administration, Saudi Arabia; and the Computer Science Department at the Faculty of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University. Her research interests include data science, artificial intelligence, machine learning, data mining, and software engineering. She is a Fellow of the U.K. Higher Education Academy (FHEA) and currently serves as a reviewer for several academic journals.

AMEL ALI ALHUSSAN is a professor at the Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University. Her research interest lies in AI and Machine Learning Optimization.

EL-SAYED M. EL-KENAWY (Senior Member, IEEE) received his B.Sc., M.Sc., and Ph.D. degrees in computer engineering from Mansoura University. He has authored over 400 papers, accumulating around 15,000 citations and an H-index of 68. His research interests include artificial intelligence, machine learning, metaheuristic optimization, engineering optimization, deep learning, and large language models.