Metaheuristic Algorithms with Applications

www.maa.reapress.com

Metaheur. Algor. Appl. Vol. 3, No. 1 (2026) 1–32.

Paper Type: Original Article

Whale Optimization Algorithm: A Critical Survey of Fundamental Principles, Modern Adaptations, Benchmarking Practices, and Applications

Saman M Almufti[1,2*], Renas Rajab Asaad[1,2], Awaz Ahmed Shaban[1], Ridwan Boya Marqas[2]

[1] Department of Information Technology, Technical College of Informatics-Akre, Akre University for Applied Sciences, Duhok, Iraq; Saman.Almofty@gmail.com; renas.beda89@gmail.com; Awaz.Amedy1@gmail.com.

[2] Department of Computer Science, College of Science, Knowledge University, Erbil, Iraq; ridwan.marqas@dpu.edu.krd.

Citation:

Received: 14 September 2025 *Revised: 23 November 2025* *Accepted: 08 January 2026*	Almufti, S. M., Asaad, R. R., Shaban, A. A., & Marqas, R. B. (2026). Whale optimization algorithm: A critical survey of fundamental principles, modern adaptations, benchmarking practices, and applications. *Metaheuristic Algorithms with Applications*, *3*(1), 1-32.

Abstract

The Whale Optimization Algorithm (WOA) is a compact swarm-intelligence optimizer inspired by the bubble-net feeding strategy of humpback whales. Since its introduction, WOA has been adopted in continuous, discrete, constrained, and multi-objective optimization because it offers a simple mathematical structure and a natural alternation between global exploration and local exploitation. Nevertheless, the expansion of WOA research has also revealed persistent limitations, including premature convergence, rapid diversity loss, sensitivity to control schedules, weak theoretical guarantees, and inconsistent empirical comparison. This expanded survey provides a critical and mechanism-oriented synthesis of WOA from its biological inspiration and canonical equations to modern adaptive, chaotic, hybrid, discrete, multi-objective, learning-assisted, and parallel variants. Unlike descriptive summaries that list modifications without explaining their search role, this paper classifies adaptations according to the algorithmic component they modify: initialization, parameter control, position updating, perturbation, representation, archiving, learning-based strategy selection, and computational acceleration. The survey further links these mechanisms to practical domains such as structural design, energy systems, Machine Learning (ML), biomedical image analysis, Wireless Sensor Networks (WSN), cloud/edge computing, cybersecurity, scheduling, and intelligent transportation. A dedicated benchmarking section proposes reproducible evaluation practices, including benchmark diversity, statistical testing, ablation analysis, constraint handling, and transparent reporting. Finally, the paper identifies open challenges in convergence theory, large-scale optimization, dynamic environments, many-objective search, and trustworthy integration with Deep Learning (DL) and surrogate modeling. The resulting discussion is intended to serve as a stronger foundation for researchers who wish to select, compare, or develop WOA-based optimizers for rigorous scientific and engineering use.

Keywords: Whale optimization algorithm, Swarm intelligence, Metaheuristic optimization, Hybrid optimization, Multi-objective optimization, Feature selection, Engineering design, Benchmarking, reproducibility.

1 | Introduction

Optimization has become a central computational requirement in engineering, computer science, operations research, artificial intelligence, energy planning, and biomedical analysis. Modern problems are rarely smooth, convex, or low-dimensional; they are frequently nonlinear, constrained, multimodal, noisy, expensive to evaluate, and subject to multiple conflicting objectives. Under these conditions, exact optimization and

Corresponding Author: Saman.Almofty@gmail.com

https://doi.org/10.48313/maa.v3i1.110

classical gradient-based methods may either become impractical or require assumptions that do not hold in real-world systems. Metaheuristic algorithms have therefore emerged as flexible search frameworks capable of obtaining high-quality approximate solutions with limited information about the structure of the objective function [1–4].

The Whale Optimization Algorithm (WOA), introduced by Mirjalili and Lewis [5], is one of the most recognizable swarm-intelligence algorithms developed during the last decade. Its search logic abstracts the bubble-net hunting behavior of humpback whales into three major operators: encircling the best solution, moving around it through a logarithmic spiral, and exploring the search space by referring to randomly selected agents. These mechanisms produce a simple optimizer with few control parameters, concise mathematical equations, and competitive performance on several numerical and engineering-design benchmarks.

The popularity of WOA is partly explained by its ease of implementation. A basic WOA code can be written using only population initialization, fitness evaluation, coefficient-vector calculation, boundary handling, and position updates. This accessibility encouraged rapid adoption in domains such as maximum power point tracking, economic load dispatch, feature selection, image segmentation, neural-network training, wireless sensor network clustering, task scheduling, and structural design. However, the same simplicity has also created a proliferation of variants whose technical contributions are not always clearly separated from routine operator replacement. This rapid diffusion has also been documented in WOA-specific review studies [6–8].

A stronger review of WOA must therefore move beyond the question of whether a new variant performs better on a limited benchmark set. It should ask which weakness the variant addresses, which part of the search process is modified, what computational cost is added, whether the improvement is statistically significant, and whether the mechanism remains useful outside the benchmark functions used by the original authors. Without this level of analysis, the literature can become an accumulation of algorithm names rather than a coherent body of optimization knowledge. This paper expands and strengthens the survey by organizing WOA research around search mechanisms, empirical evidence, and practical relevance. It distinguishes between fundamental design principles, modification strategies, application-driven adaptations, and methodological requirements for fair benchmarking. The analysis is especially focused on the relationship between exploration and exploitation, because almost every successful WOA variant can be interpreted as an attempt to modify this relationship either directly or indirectly.

The contributions of this expanded paper are fivefold: 1) it provides a complete and corrected presentation of the canonical WOA mathematical model, 2) it offers a mechanism-based taxonomy of WOA adaptations, 3) it critically compares the purposes and limitations of major improvement families, 4) it maps WOA variants to application domains and explains why particular variants are suitable for particular classes of problems, and 5) it proposes a reproducible benchmarking framework and identifies research directions that remain insufficiently explored. The rest of the paper is organized as follows. Section 2 presents the review scope and guiding research questions. Section 3 introduces the foundations of metaheuristic optimization. Section 4 explains the original WOA model. Section 5 develops a detailed taxonomy of WOA variants. Section 6 discusses application domains. Section 7 focuses on benchmarking and reproducibility. Section 8 provides a critical discussion of limitations and evidence. Section 9 proposes future research directions. Sections 10-13 provide design guidelines, quality assessment, domain-specific recommendations, and a research roadmap, while Section 14 concludes the paper.

2 | Review Scope and Analytical Method

This paper is designed as a structured narrative survey rather than a formal bibliometric or Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA)-based systematic review. The objective is to synthesize the technical evolution of WOA, clarify the function of major enhancements, and provide guidance for researchers who wish to select, compare, or develop WOA variants. Accordingly, the emphasis is placed on mechanism, methodological quality, and practical interpretation rather than publication counts alone.

The reviewed body of work can be divided into four evidence layers. The first layer contains foundational optimization and metaheuristic literature, including the original WOA paper and related population-based algorithms such as Particle Swarm Optimization (PSO), Genetic Algorithms (GA), simulated annealing, and gravitational search algorithms. The second layer contains surveys and systematic reviews that summarize the development of WOA and its variants. The third layer contains algorithmic papers that introduce modifications, hybridizations, or theoretical analyses. The fourth layer contains application studies in engineering, power systems, Machine Learning (ML), image processing, communications, and other domains. The foundational layer includes classical metaheuristics that remain common baselines for WOA evaluation [9–11].

Inclusion was based on conceptual relevance and methodological clarity. A study was considered relevant when it introduced a WOA variant, analyzed the behavior of WOA, compared WOA with established optimizers, used WOA for a well-defined application, or provided a significant survey or taxonomy. Studies were treated cautiously when they reported superiority using only one dataset, omitted parameter settings, failed to conduct repeated independent runs, or did not provide statistical testing. Such studies may still be useful for identifying application trends, but they should not be used as strong evidence of general algorithmic superiority. The analytical method used in this survey is component-based. Instead of classifying variants only by their names, each modification is examined according to the part of the algorithm that it affects: initialization, coefficient scheduling, randomization, local search, representation, archive management, constraint handling, or computational parallelization. This method makes it possible to compare apparently different variants that actually target the same search deficiency. For example, chaotic initialization, opposition-based learning, and Latin hypercube sampling are different techniques, but all attempt to improve initial diversity and coverage. The survey also separates algorithmic contribution from application contribution. A paper that applies WOA to photovoltaic control may be important for renewable energy, even if it does not introduce a new optimizer. Conversely, a paper that introduces an advanced hybrid WOA may not demonstrate practical value unless it is evaluated under realistic objective functions, constraints, and computational budgets. Strong future research should ideally contribute to both dimensions: it should modify the optimizer for a clear reason and validate the modification in a credible problem context.

Table 1. Guiding review questions used to strengthen the expanded survey.

Review Question	Analytical Focus	Expected Strengthening Effect
RQ1: What makes the canonical WOA attractive?	Mathematical simplicity, few parameters, intuitive exploration-exploitation structure, and ease of implementation	Clarifies why WOA became widely adopted
RQ2: What weaknesses motivate WOA variants?	Premature convergence, diversity loss, sensitivity to parameter schedules, weak local search, and representation limits	Explains why modification is necessary
RQ3: How should WOA variants be classified?	By the search component modified rather than by the name of the hybrid algorithm	Produces a deeper and less repetitive taxonomy
RQ4: Where is WOA most useful?	Domains with black-box, nonlinear, multimodal, or constrained objectives where derivative information is unavailable	Links algorithm choice to problem structure
RQ5: How should WOA be evaluated?	Repeated runs, diverse benchmarks, fair baselines, statistical tests, ablation analysis, and reproducible code	Improves methodological credibility

3 | Foundations of Metaheuristic Search

Metaheuristics are general-purpose optimization strategies designed to explore large search spaces without relying on strict assumptions such as convexity, differentiability, or linearity. They are especially valuable when objective functions are discontinuous, nonseparable, noisy, multimodal, or computationally expensive. Unlike exact methods, metaheuristics do not guarantee global optimality in finite time for arbitrary problems; instead, they aim to provide robust approximations under realistic computational limits.

Population-based metaheuristics maintain a set of candidate solutions, often called agents, particles, individuals, or search points. This population enables simultaneous exploration of multiple regions of the search space and reduces dependence on a single initial solution. Information may be exchanged through global-best memory, neighborhood structures, selection pressure, social learning, or archive-based mechanisms. WOA belongs to this population-based category because it updates a group of whales according to the current best candidate and stochastic search rules.

A central principle in metaheuristic design is the balance between exploration and exploitation. Exploration refers to wide-ranging search intended to discover promising regions and avoid premature commitment to local optima. Exploitation refers to intensifying search around high-quality solutions in order to refine objective values and accelerate convergence. A search process that overemphasizes exploration may waste evaluations, whereas one that overemphasizes exploitation may converge too early. The WOA model explicitly encodes this balance through the coefficient vector A and the probability-based selection between encircling and spiral motion.

The no-free-lunch perspective is also relevant. No optimizer can outperform all others on all possible problems when averaged over the space of all objective functions. Therefore, claims that one WOA variant is universally superior should be interpreted carefully. A more scientifically useful claim is conditional: a variant may be more suitable for high-dimensional functions, discrete feature selection, constrained engineering design, expensive simulations, or multi-objective trade-off problems. This conditional interpretation is essential for meaningful algorithm selection.

Constraint handling is another fundamental issue. Many engineering and scheduling problems include inequality constraints, equality constraints, boundary constraints, or discrete feasibility rules. WOA, like many continuous metaheuristics, requires additional mechanisms to deal with infeasible solutions. Common strategies include penalty functions, feasibility rules, repair operators, decoders, constraint-preserving representations, and multi-objective treatment of constraint violation. The choice of strategy can strongly affect results and should be reported clearly.

Computational complexity is usually dominated by objective-function evaluation rather than position updates. For a population of N whales, dimensionality D, and maximum iteration count T, the update cost of canonical WOA is approximately O(TND), while the total cost is O(TN(Cf + D)), where Cf denotes the cost of one fitness evaluation. In expensive engineering simulations, Cf may be orders of magnitude larger than the update equations. This explains why Surrogate-Assisted WOA (SA–WOA) and Parallel WOA (PWOA) have become increasingly relevant.

The design of metaheuristic experiments must also recognize stochastic variability. Because random initialization and random coefficients influence the trajectory of the search, a single run cannot represent algorithm performance. Reliable comparison requires repeated independent runs, appropriate summary statistics, and nonparametric significance tests when normality cannot be assumed. This requirement is not a minor methodological detail; it is central to the credibility of WOA research.

Table 2. Position of WOA within the broader metaheuristic landscape.

Algorithm Family	Search Metaphor	Typical Strength	Typical Weakness
Evolutionary algorithms	Selection, recombination, and mutation inspired by biological evolution	Strong population diversity and flexible encoding	May require several operators and parameter tuning
Swarm intelligence	Collective behavior of animals or agents, including particles, ants, wolves, and whales	Simple social learning and effective global search	Risk of premature convergence around a dominant leader
Physics-based methods	Physical phenomena such as annealing, gravity, electromagnetism, or equilibrium	Useful stochastic transitions and acceptance rules	Metaphor may not always translate into robust search
Human/social-inspired methods	Teaching, learning, competition, cooperation, or social behavior	Intuitive learning structures and adaptive behavior	Can become problem-dependent or hard to justify theoretically
Hybrid methods	Combination of two or more search mechanisms	Potentially improved balance between global and local search	Additional complexity and risk of unfair comparison

4 | Whale Optimization Algorithm

4.1 | Biological Inspiration and Search Interpretation

The biological inspiration of WOA is the bubble-net feeding behavior of humpback whales. In this hunting strategy, whales create circular or spiral bubbles around prey and move in a coordinated pattern that restricts prey movement. The algorithm abstracts this behavior by assuming that the best solution currently known represents the prey, while the remaining candidate solutions represent whales that update their positions relative to that prey. Although the biological metaphor is useful pedagogically, the optimization meaning is more important. Encircling prey corresponds to attraction toward the best solution; spiral movement corresponds to nonlinear local exploitation around the best solution; and random search corresponds to exploration guided by another randomly chosen population member. The performance of WOA depends on the dynamic interaction among these mechanisms rather than the metaphor alone.

At each iteration, WOA updates coefficient vectors that determine whether a whale moves toward the current best solution or explores around a random solution. The linearly decreasing control parameter a gradually changes the algorithm from exploration-oriented behavior to exploitation-oriented behavior. This schedule is simple, but it assumes that early iterations should explore and later iterations should exploit. Many improved variants challenge this assumption by using nonlinear, chaotic, adaptive, entropy-based, or learning-driven schedules.

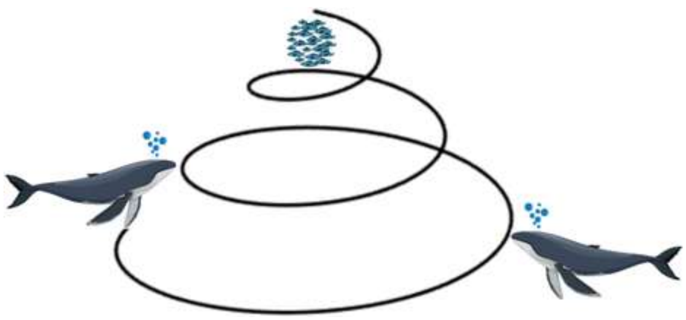

Fig. 1. Conceptual hunting strategy that motivates the WOA search model.

4.2 | Mathematical Model

Let X(t) denote the position vector of a whale at iteration t, and let X*(t) denote the best solution found so far. WOA models the distance between a whale and the current best solution using coefficient vector C as follows:

$$D = | C \cdot X^*(t) - X(t) |. \tag{1}$$

The position of the whale is then updated toward the best solution using coefficient vector A:

$$X(t + 1) = X^*(t) - A \cdot D. \tag{2}$$

The coefficient vectors A and C are calculated as:

$$A = 2a \cdot r - a, C = 2r, \tag{3}$$

where r is a random vector in [0,1], and a decreases linearly from 2 to 0 over the iterations. When |A| < 1, the search tends to exploit the region around the best solution; when |A| >= 1, the search is encouraged to move away and explore more broadly. This simple condition is one of the main reasons WOA is easy to understand and implement.

The spiral updating mechanism is used to model bubble-net attacking. The distance between the whale and the best solution is defined as:

$$D' = | X^*(t) - X(t) |. \tag{4}$$

The spiral position update is then expressed as:

$$X(t + 1) = D' \cdot e\hat{}(bl) \cdot \cos(2\pi l) + X^*(t), \tag{5}$$

where b is a constant controlling the logarithmic spiral shape and l is a random number in [-1,1]. In the canonical algorithm, a probability p determines whether a whale follows the encircling mechanism or the spiral updating mechanism. When p < 0.5, encircling is applied; otherwise, spiral updating is applied. Exploration occurs when |A| >= 1, in which case the position is updated relative to a randomly selected whale Xrand:

$$D = | C \cdot Xrand - X(t) |, X(t + 1) = Xrand - A \cdot D. \tag{6}$$

Boundary control is required after position updating. The most common approach clips values that exceed lower or upper bounds, although reflection, random reinitialization, and repair operators may be preferable for some constrained problems. The choice of boundary control is rarely discussed in detail, yet it can influence convergence behavior, especially in high-dimensional spaces where many coordinates may leave the feasible region simultaneously.

4.3 | Canonical Pseudocode and Computational Cost

Table 3. Canonical WOA procedure in compact pseudocode form.

Step	Operation
1	Initialize N whale positions uniformly within the search bounds.
2	Evaluate the fitness value of each whale and identify the best solution X*.
3	For each iteration t = 1, 2, ..., T, update a, A, C, l, and p.
4	For each whale, choose encircling, spiral updating, or random exploration according to p and |A|.
5	Apply boundary control and evaluate the new candidate solution.
6	Update X* if a better feasible solution is found.
7	Return X* after the termination criterion is satisfied.

The computational cost of WOA is primarily determined by the population size, dimensionality, iteration count, and objective-function evaluation cost. The position update itself is linear in population size and dimensionality. However, in engineering optimization, medical image processing, or neural-network training, each evaluation may involve simulation, segmentation, model training, or cross-validation. In such cases, reducing the number of evaluations or parallelizing them may be more important than refining the mathematical update equation.

The memory requirement is modest because WOA only needs to store the population matrix, fitness values, and the current best solution. This makes the algorithm attractive for embedded or resource-constrained applications, although variants that use external archives, surrogate models, multiple subpopulations, or historical memory may increase storage requirements.

4.4 | Strengths and Weaknesses of the Original Whale Optimization Algorithm

Table 4. Practical strengths and limitations of the canonical WOA.

Aspect	Strength	Weakness or Risk
Simplicity	Few equations and limited parameter burden make WOA easy to implement	Simplicity may limit flexibility in complex landscapes
Exploration-exploitation balance	The a coefficient offers an intuitive transition from exploration to exploitation	A fixed linear schedule may not match the difficulty of all problems
Local exploitation	Spiral updating supports refined search around the best solution	Excessive attraction to the best solution can reduce diversity quickly
Problem independence	WOA can be applied to black-box functions without gradients	Discrete, binary, and constrained problems require additional representation mechanisms
Computational cost	Update equations are inexpensive	Objective evaluation may dominate cost in real-world simulations

5 | Taxonomy of Whale Optimization Algorithm Modifications and Hybridizations

WOA variants can be interpreted as attempts to correct one or more limitations of the canonical algorithm. The most frequent motivations are to maintain population diversity, accelerate convergence, improve local exploitation, escape local optima, adapt parameters automatically, handle discrete representations, manage multiple objectives, or reduce computational cost. A mechanism-based taxonomy is more useful than a name-based taxonomy because it reveals recurring design patterns across apparently different variants. Recent hybrid and multi-strategy developments show that this trend is continuing [12], [13].

The following subsections organize WOA improvements according to the search component that is modified. This organization helps researchers select an appropriate variant for a problem. For example, feature selection typically requires binary representation and sparsity-aware fitness design; expensive engineering optimization benefits from surrogate assistance; multi-objective design requires Pareto archiving; and high-dimensional continuous optimization may benefit from decomposition or multi-population cooperation.

5.1 | Initialization and Diversity Enhancement

Initialization affects the first distribution of candidate solutions and can influence convergence speed, especially when the budget of fitness evaluations is limited. Uniform random initialization is simple but may leave large regions of the search space under-sampled. Improved initialization strategies include chaotic maps, opposition-based learning, quasi-random sequences, Latin hypercube sampling, orthogonal design, and clustering-based initialization.

Diversity enhancement may also be introduced during the search rather than only at the beginning. Random reinitialization of stagnant agents, crowding measures, entropy-based feedback, or multi-swarm separation can prevent all whales from collapsing around a suboptimal leader. These techniques are particularly relevant for multimodal functions and high-dimensional spaces.

5.2 | Adaptive and Self-Adaptive Parameter Control

The canonical WOA uses a linearly decreasing parameter a. Although simple, this deterministic schedule does not respond to search feedback. Adaptive WOA variants replace it with nonlinear, exponential, sinusoidal, chaotic, fitness-based, entropy-based, or self-adaptive schedules. The underlying assumption is that the optimizer should adapt its behavior according to progress rather than iteration count alone. Adaptive and improved WOA studies commonly modify this schedule to reduce stagnation and accelerate convergence [12], [13].

Self-adaptive control can be based on population diversity, improvement rate, stagnation duration, or distance to the best solution. When diversity is high, the algorithm can exploit more aggressively; when diversity is low and progress stagnates, it can reintroduce exploration. The main risk is that adaptive control may add new hyperparameters that themselves require tuning. A strong adaptive WOA paper should therefore include ablation studies showing that the adaptation rule, not hidden parameter tuning, causes the observed improvement.

5.3 | Chaotic, Stochastic, and Perturbation-Based Whale Optimization Algorithm

Chaotic maps are commonly used to improve randomness and diversity. Logistic, tent, sine, Gauss, and Chebyshev maps can be used for initialization, coefficient generation, or perturbation. Chaotic behavior may help WOA avoid repetitive search patterns, but the effectiveness of a particular map depends on the problem and implementation details. Therefore, chaotic WOA variants should compare several maps or justify why a specific map is selected. Chaotic WOA is a representative example of this diversity-oriented design direction [14].

Perturbation-based variants use Gaussian mutation, Cauchy mutation, Lévy flights, random walks, or heavy-tailed distributions to move agents beyond the current attraction region. Gaussian mutation supports local refinement, whereas Cauchy and Lévy mechanisms produce occasional long jumps that may escape local optima. These mechanisms are valuable when the standard encircling process collapses too rapidly.

5.4 | Hybrid Whale Optimization Algorithm with other Metaheuristics

Hybridization is one of the dominant trends in WOA research. WOA has been combined with PSO, Differential Evolution (DE), GA, Grey Wolf Optimizer (GWO), Sine Cosine Algorithm (SCA), Artificial Bee Colony (ABC), Ant Colony Optimization (ACO), simulated annealing, firefly algorithm, harmony search, teaching-learning-based optimization, and many other optimizers. The purpose is usually to combine WOA exploration with another algorithm's exploitation, mutation, recombination, memory, or acceptance mechanism. Several improved WOA models use strategic adjustment or multi-strategy control to strengthen the global-local search balance [15], [16].

Hybridization should not be treated as automatically beneficial. A hybrid algorithm may outperform each component because it genuinely combines complementary mechanisms, but it may also outperform them simply because it uses more operators, more parameters, or more computational effort. Fair comparison should therefore control the number of objective evaluations and include ablation analysis showing the contribution of each component.

5.5 | Binary, Discrete, and Combinatorial Whale Optimization Algorithm

The original WOA is designed for continuous search spaces. Many practical problems, however, are binary, integer, permutation-based, or mixed discrete-continuous. Examples include feature selection, unit commitment, task scheduling, routing, knapsack problems, and sensor placement. Binary WOA usually transforms continuous positions into binary decisions using S-shaped, V-shaped, or other transfer functions. Discrete WOA may use problem-specific encodings, decoders, swap operators, or repair rules. Binary and feature-selection-oriented WOA variants demonstrate the importance of representation design in discrete search [17–20].

The critical issue in discrete WOA is representation. A poor representation can produce infeasible or meaningless solutions even if the update rule is mathematically elegant. Therefore, discrete WOA studies should explain how candidate solutions are encoded, how updates are mapped to valid decisions, and how feasibility is preserved or repaired. In feature selection, for example, the fitness function should balance classification accuracy with the number of selected features, rather than optimizing accuracy alone.

5.6 | Multi-Objective and Many-Objective Whale Optimization Algorithm

Many real optimization problems involve conflicting objectives, such as cost and reliability, accuracy and model complexity, energy consumption and latency, or structural weight and safety. Multi-objective WOA variants typically incorporate Pareto dominance, external archives, crowding distance, grid-based selection, leader selection, and diversity-preserving mechanisms. The best whale is no longer a single solution but may be selected from an archive of nondominated candidates. Multi-objective WOA research has therefore developed archive-based and Pareto-guided mechanisms for trade-off optimization [21–23].

Many-objective optimization, usually involving more than three objectives, remains more difficult. Pareto dominance loses selection pressure as the number of objectives increases, and crowding measures become less informative. Future WOA research should investigate decomposition-based, reference-vector, indicator-based, and preference-driven many-objective frameworks rather than simply extending two-objective methods.

5.7 | Learning-Assisted, Surrogate-Assisted, and Parallel Whale Optimization Algorithm

Recent work increasingly connects WOA with ML. WOA can optimize neural-network weights, select features, tune hyperparameters, or select model architectures. Conversely, ML can assist WOA by predicting promising regions, controlling parameters, selecting operators, or approximating expensive objective functions. Reinforcement learning is particularly relevant for strategy selection because the optimizer can treat exploration and exploitation operators as actions.

SA–WOA is useful when objective evaluations are expensive. A surrogate model approximates the objective function using previously evaluated samples and guides the search toward promising candidates. The main challenge is managing model uncertainty: if the surrogate is inaccurate, the optimizer may converge to an artifact of the approximation. PWOA addresses computational cost by distributing fitness evaluations or maintaining multiple subpopulations that communicate periodically.

Table 5. Mechanism-based taxonomy of major WOA variant families.

Variant Family	Primary Mechanism	Problem Addressed	Main Caution
Adaptive WOA	Dynamic control of a, A, C, or probability p	Fixed schedules may not fit all landscapes	Adaptation may add new parameters
Chaotic WOA	Chaotic maps for initialization or coefficient generation	Low diversity and repetitive random patterns	Map selection can be arbitrary
Opposition-based WOA	Evaluates opposite candidates to improve coverage	Poor initial sampling and slow early progress	Extra evaluations must be counted fairly
Lévy-flight WOA	Heavy-tailed jumps added to position updates	Local optima and narrow attraction basins	Excessive jumps may destabilize convergence
Mutation-Based WOA (MBWOA)	Gaussian, Cauchy, or differential perturbation	Stagnation and weak local refinement	Mutation scale requires careful design
WOA-PSO	Combines whale movement with velocity/social learning	Need for stronger exploitation or social memory	Can inherit PSO premature convergence
WOA-DE	Uses mutation and crossover from DE	Need for diversity and vector-difference exploration	More parameters and higher implementation complexity
WOA-GA	Introduces selection, crossover, and mutation	Discrete representations and diversity maintenance	Operator design must fit encoding
Binary WOA	Transfer functions convert positions into binary decisions	Feature selection, knapsack, and on/off decisions	Transfer function strongly affects results
Multi-objective WOA	Pareto archive and leader selection	Conflicting objectives and trade-off analysis	Archive maintenance can be costly
SA–WOA	Approximate expensive objective evaluations	Simulation-heavy engineering optimization	Surrogate error can mislead the search
PWOA	Distributed evaluations or subpopulations	Large-scale or expensive optimization	Communication and synchronization overhead

6 | Application Domains of Whale Optimization Algorithm

WOA has been applied across a wide range of domains because many real-world problems can be formulated as black-box optimization tasks. However, the success of WOA in a domain depends on the suitability of its representation, constraint-handling method, and evaluation protocol. This section discusses major application areas and highlights the specific reasons WOA is attractive in each area.

A critical application review should avoid treating all successful case studies as equivalent. Some applications demonstrate only that WOA can be used, whereas others show that WOA is particularly well suited to the problem structure. Strong evidence requires comparison against domain-specific methods, sensitivity analysis, and validation on realistic data. The following subsections therefore emphasize both opportunities and methodological cautions.

6.1 | Structural and Mechanical Engineering

Structural engineering problems such as truss design, pressure-vessel design, welded-beam design, spring design, and frame optimization are common testbeds for WOA. These problems often include nonlinear constraints, safety limits, material bounds, and competing design requirements. WOA is attractive because it can handle nonlinear objective functions and constraints without gradients. Hybrid WOA variants are especially useful when the feasible region is narrow or the optimum lies near active constraints.

In practical structural optimization, the objective is not merely to minimize cost or weight; the design must satisfy stress, displacement, buckling, frequency, and manufacturability constraints. Therefore, the quality of constraint handling is often more important than the choice of search metaphor. Future structural WOA studies should report feasibility rates, constraint violation statistics, and sensitivity to penalty coefficients rather than only the best objective value.

6.2 | Energy Systems and Power Engineering

Energy and power systems have become a major application area for WOA. Problems include maximum power point tracking in photovoltaic systems, economic load dispatch, optimal power flow, unit commitment, capacitor placement, load-frequency control, demand response, microgrid scheduling, and renewable generation forecasting. These problems are nonlinear, constrained, and often affected by uncertainty in load, weather, and market conditions.

WOA is useful in this domain because it can optimize control parameters or schedules under nonlinear power-balance constraints. However, energy applications require careful validation under dynamic operating conditions. For example, MPPT algorithms should be tested under partial shading and rapid irradiance changes, while economic dispatch methods should include valve-point effects, prohibited operating zones, emission constraints, and renewable uncertainty when relevant.

6.3 | Machine Learning, Feature Selection, and Data Mining

WOA has been widely applied to feature selection, clustering, classification, hyperparameter tuning, neural-network training, and deep-learning optimization. In feature selection, each whale represents a subset of features, and the objective function usually combines classification performance with feature-count reduction. Binary WOA variants are common because feature selection is naturally a binary decision problem.

The main methodological challenge in machine-learning applications is avoiding overfitting. If WOA selects features or hyperparameters using the same data used for final testing, the reported accuracy may be overly optimistic. Strong studies should use nested cross-validation, independent test sets, or repeated stratified validation. They should also report selected feature counts, computational cost, and stability of selected features across runs.

6.4 | Biomedical, Bioinformatics, and Image Processing

Biomedical applications include medical image segmentation, disease classification, gene selection, protein-structure prediction, drug-target interaction modeling, and biomedical signal analysis. WOA is attractive because these problems often involve high-dimensional feature spaces, nonlinear relationships, and noisy data. In image segmentation, WOA can optimize threshold values, clustering centers, or model parameters. In bioinformatics, it can search large combinatorial spaces such as gene subsets.

The risk in biomedical optimization is that algorithmic performance may be reported without sufficient clinical or biological validation. A segmentation result should be evaluated using metrics such as Dice coefficient, sensitivity, specificity, boundary distance, and visual assessment by domain experts when possible. Gene-selection studies should consider biological interpretability and reproducibility, not only classification accuracy.

6.5 | Wireless Sensor Networks, Internet of Things, Cloud, and Edge Computing

Wireless Sensor Networks (WSN) and IoT systems often require energy-efficient clustering, node deployment, routing, coverage optimization, and lifetime maximization. WOA can optimize cluster-head selection, transmission schedules, or deployment positions. In cloud and edge computing, WOA has been used for task scheduling, virtual-machine placement, resource provisioning, computation offloading, and load balancing.

These applications are characterized by multiple objectives such as energy consumption, latency, reliability, throughput, and service quality. Therefore, multi-objective WOA and constrained WOA are often more appropriate than the canonical single-objective form. Researchers should evaluate algorithms using realistic workload traces or network simulations and should report not only objective values but also runtime overhead and scalability.

6.6 | Cybersecurity, Scheduling, Logistics, and Other Domains

WOA has also been applied to intrusion detection, cryptographic key generation, image encryption, test-case prioritization, job-shop scheduling, vehicle routing, supply-chain planning, portfolio optimization, hydrological modeling, transportation control, and UAV path planning. These domains often involve discrete choices, time-dependent constraints, or multiple conflicting performance indicators. Hence, they typically require customized WOA variants rather than direct use of the continuous algorithm.

Table 6. Mapping between WOA application domains, suitable variants, and evaluation criteria.

Domain	Typical WOA Formulation	Recommended Variant or Mechanism	Key Evaluation Criteria
Structural design	Continuous constrained minimization of weight, cost, or stress	Hybrid WOA, adaptive penalties, repair operators	Best/mean cost, feasibility, constraint violation, safety margin
Power systems	Nonlinear scheduling or control with operational constraints	Adaptive WOA, hybrid WOA-DE/PSO, multi-objective WOA	Fuel cost, emission, voltage stability, convergence, reliability
Feature selection	Binary subset optimization with classifier wrapper	Binary WOA, transfer functions, sparsity penalty	Accuracy, feature count, F-score, stability, runtime
Image segmentation	Threshold or cluster-center optimization	Chaotic WOA, Lévy WOA, hybrid local search	Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), Dice score, threshold quality, visual clarity
WSN/IoT	Cluster-head selection, routing, or deployment	Multi-objective WOA, energy-aware WOA	Lifetime, coverage, delay, packet delivery, energy use
Cloud/edge computing	Task allocation and offloading	Parallel or multi-objective WOA	Makespan, latency, energy, cost, resource utilization
Cybersecurity	Feature selection, rule tuning, or key generation	Binary WOA, hybrid WOA-ML	Detection rate, false positives, robustness, overhead
Scheduling/logistics	Permutation or integer optimization	Discrete WOA, decoders, repair operators	Makespan, lateness, cost, feasibility, scalability

7 | Benchmarking, Statistical Evaluation, and Reproducibility

Benchmarking is essential for evaluating any WOA variant, but benchmark design is often the weakest part of metaheuristic research. A fair comparison should use diverse test functions, realistic dimensions, consistent stopping criteria, equal numbers of fitness evaluations, repeated independent runs, and appropriate statistical tests. Reporting only the best result is insufficient because it hides variability and may reward lucky runs.

Benchmark functions should include unimodal functions for exploitation assessment, multimodal functions for local-optima avoidance, hybrid and composition functions for landscape complexity, rotated or shifted functions for nonseparability, and constrained engineering problems for practical relevance. A variant that performs well only on simple unimodal functions may be effective at local exploitation but weak at global exploration. Conversely, a variant that performs well on multimodal functions but converges slowly may be unsuitable for expensive applications. Modern benchmark construction also emphasizes shifted, rotated, and structurally challenging functions [24].

Performance indicators should include best, mean, median, standard deviation, convergence curves, success rate, and runtime. For multi-objective variants, hypervolume, Inverted Generational Distance (IGD), spacing, spread, and archive diversity may be needed. For constrained problems, feasible rate and total constraint violation should be reported. For feature selection, accuracy should be accompanied by feature-count reduction and stability.

Statistical analysis should match the stochastic nature of WOA. The Wilcoxon signed-rank test, Friedman test, Holm post-hoc procedure, and effect-size measures are commonly used in metaheuristic comparison. Statistical significance alone is not enough; practical significance should also be discussed. A tiny improvement that requires twice the computational cost may not be meaningful in real applications.

Ablation analysis is particularly important for hybrid WOA. When a variant includes chaotic initialization, adaptive control, Lévy flights, and local search, the paper should test each component separately. Otherwise, it is impossible to know which component produced the improvement. Ablation also prevents unnecessary complexity by identifying operators that contribute little or only help on narrow benchmark classes.

Reproducibility requires detailed reporting. Authors should disclose population size, maximum iterations, stopping criteria, dimensionality, bounds, penalty functions, random seeds, software environment, hardware, dataset splits, and all parameter values. When source code cannot be released, pseudocode should be precise enough to reimplement the method. Ambiguous descriptions such as “parameters were set experimentally” weaken the credibility of the study.

Table 7. Recommended benchmarking and reproducibility checklist for WOA studies.

Benchmarking Element	Minimum Recommended Practice	Reason
Runs	At least 30 independent runs for stochastic comparison	Captures variability and reduces dependence on a lucky run
Stopping criterion	Equal fitness-evaluation budget across algorithms	Prevents hybrids from gaining unfair advantage through extra evaluations
Metrics	Report best, mean, median, standard deviation, and convergence curves	Shows both performance level and reliability
Statistics	Use nonparametric tests and post-hoc analysis when comparing many algorithms	Supports evidence-based claims of superiority
Parameters	Disclose all settings and tuning strategy	Allows replication and fair interpretation
Datasets	Use training/validation/test separation in ML tasks	Reduces overfitting and inflated accuracy
Constraints	Report feasibility rate and violation values	Distinguishes good feasible solutions from infeasible numerical artifacts
Ablation	Test each added component of a hybrid variant	Identifies whether complexity is justified

8 | Critical Discussion

The WOA literature demonstrates that the algorithm is flexible and widely applicable, but it also reveals several recurring weaknesses in how metaheuristic contributions are framed. Many papers introduce a new WOA variant by combining several operators from other algorithms, then evaluate it on a limited benchmark set. Although such studies may produce improved numerical results, they do not always clarify the mechanism responsible for the improvement or the conditions under which the variant should be preferred.

A major issue is the overuse of broad claims. Statements such as "the proposed WOA is superior to all existing algorithms" are rarely defensible. A more accurate statement would specify the benchmark suite, dimensionality, objective type, constraint structure, and computational budget under which the variant performed well. Algorithmic superiority is context-dependent, and the survey should encourage precise claims rather than universal statements.

Another limitation is inadequate comparison with strong baselines. Some WOA papers compare against outdated algorithms or weak parameter settings. A strong evaluation should compare against both classical baselines and recent competitive algorithms, including non-WOA methods. Otherwise, the study may show only that a variant is better than a weak reference, not that it represents a meaningful advance.

The metaphor-driven nature of metaheuristics also requires caution. Biological inspiration can motivate algorithm design, but performance must be justified through mathematical behavior, empirical evidence, and problem suitability. A new metaphor is not itself a contribution unless it leads to a distinct and effective search mechanism. For WOA, the most valuable improvements are those that can be explained in terms of diversity, convergence, perturbation, representation, or computational efficiency.

Despite these criticisms, WOA remains valuable. Its compact structure makes it an excellent base algorithm for experimentation, teaching, and rapid prototyping. It is also suitable for hybridization because its main operators are easy to replace or augment. The challenge for future research is not simply to create more variants, but to create better-justified variants supported by transparent evidence.

In practice, WOA should be selected when the problem is black-box, nonlinear, derivative-free, and moderately expensive, and when a population-based search is appropriate. It may be less suitable when exact mathematical structure is available, when constraints require specialized solvers, when real-time guarantees are strict, or when the problem is extremely high-dimensional without decomposition. In such cases, WOA may still be useful as part of a hybrid system, but not necessarily as the primary solver.

Table 8. Recurrent weaknesses in WOA studies and recommended corrections.

Common Weakness in WOA Papers	Why it Matters	How to Strengthen Future Work
Only best values are reported	Best values hide instability and random luck	Report mean, median, deviation, convergence, and all-run statistics
Unclear parameter tuning	Results may depend on hidden manual adjustment	State tuning protocol and use equal effort across algorithms
Too few benchmark functions	Generalization cannot be assessed	Use diverse landscapes and real-world cases
No ablation analysis	Contribution of each component is unknown	Evaluate the base algorithm plus each added operator
No runtime or evaluation budget	Improvement may come from extra computation	Normalize comparisons by fitness evaluations and report runtime
Weak constraint reporting	Infeasible solutions may appear strong	Report feasibility rate and violation magnitude
Overgeneralized claims	Misleads readers about applicability	State the domain and conditions where improvement is observed

9 | Future Research Directions

9.1 | Theoretical Analysis and Convergence Guarantees

Theoretical understanding of WOA remains less developed than its empirical applications. Future research should analyze convergence under simplified assumptions, stability of position updates, diversity decay, and the relationship between coefficient schedules and search trajectories. Markov-chain analysis, dynamical-systems interpretation, runtime analysis, and stochastic-process modeling may provide useful foundations.

Theory does not need to replace empirical testing, but it can guide algorithm design. For example, if analysis shows that diversity collapses too quickly under certain parameter schedules, adaptive or restart mechanisms can be designed more systematically. Similarly, understanding boundary effects may improve performance in constrained domains.

9.2 | Large-Scale and High-Dimensional Optimization

High-dimensional optimization remains one of the most difficult challenges for WOA. As dimensionality increases, the search space grows rapidly, distances become less informative, and population diversity becomes harder to maintain. Future WOA variants should investigate cooperative coevolution, variable grouping, subspace search, dimensionality reduction, covariance-informed movement, and adaptive population sizing.

Large-scale problems also require careful evaluation. A variant tested only on 30-dimensional functions may not scale to hundreds or thousands of variables. Future studies should include dimensionality-scaling experiments and analyze how runtime, convergence, and solution quality change with problem size.

9.3 | Dynamic, Noisy, and Uncertain Environments

Many practical optimization problems are dynamic: loads change, traffic flows vary, sensors fail, renewable generation fluctuates, and cyber threats evolve. Canonical WOA assumes a fixed objective function, so dynamic environments require memory, change detection, prediction, and re-diversification. Future Dynamic WOA (DWOA) variants should include mechanisms that detect environmental changes and adapt without restarting the entire search from scratch.

Noisy objectives require additional robustness. Re-evaluation, averaging, uncertainty modeling, and noise-resistant selection may be necessary. In expensive noisy problems, surrogate models should incorporate uncertainty estimates rather than providing only point predictions.

9.4 | Constrained, Multi-Objective, and Many-Objective Whale Optimization Algorithm

Future constrained WOA research should move beyond simple static penalty functions. Adaptive penalties, feasibility rules, repair operators, decoders, and constraint-domination principles can produce more reliable feasible solutions. Equality constraints remain especially challenging and require careful numerical tolerance management.

Multi-objective WOA should focus on both convergence and diversity of the Pareto front. Many-objective WOA requires new leader-selection mechanisms because conventional Pareto dominance becomes less discriminating as objectives increase. Preference-based and decomposition-based methods may help maintain selection pressure while producing interpretable trade-offs.

9.5 | Explainable and Learning-Guided Whale Optimization Algorithm

Learning-guided WOA is promising, but it should be explainable and robust. Reinforcement learning can select operators, tune parameters, or decide when to restart, but the learned policy must be tested across different problem classes. Otherwise, the optimizer may overfit to a benchmark distribution. Explainable strategy-selection models could show when exploration, exploitation, mutation, or local search is chosen and why.

Deep Learning (DL) can also interact with WOA as an application target. WOA may tune hyperparameters, select architectures, or optimize feature subsets. However, these studies should follow rigorous machine-learning validation protocols, including independent test sets, repeated runs, and comparison with established hyperparameter optimization methods such as Bayesian optimization and random search.

9.6 | Open Science, Benchmarks, and Software Quality

The long-term value of WOA research will depend on reproducibility. Future studies should release source code, benchmark scripts, datasets or dataset links, parameter files, random seeds, and raw experimental results. Standardized repositories would allow researchers to compare new variants against a common baseline rather than repeatedly reimplementing algorithms with inconsistent details.

Software quality is also important. Many metaheuristic implementations contain boundary-handling errors, inconsistent evaluation budgets, or hidden random-seed dependencies. Well-tested open-source WOA libraries would improve reliability and make it easier to transfer algorithms into applied engineering and industrial contexts.

10 | Design Guidelines for Strong Whale Optimization Algorithm Variants

A stronger WOA variant should be designed from an explicit diagnosis of the search problem. The developer should first identify whether the main difficulty is poor initial coverage, premature convergence, weak exploitation, infeasible solutions, high dimensionality, expensive evaluation, discrete representation, or multiple conflicting objectives. Only after this diagnosis should an operator be added. This principle prevents arbitrary hybridization and encourages mechanism-driven algorithm design.

If the weakness is poor initial coverage, the most appropriate interventions are initialization and diversity strategies. Chaotic maps, opposition-based learning, quasi-random sequences, clustering-based seeding, and Latin hypercube sampling can improve early coverage. However, these methods should be evaluated with the same number of objective evaluations as the baseline. If opposition-based learning doubles the number of initial evaluations, the comparison should either count those evaluations or provide the baseline with the same budget.

If the weakness is premature convergence, the solution is not necessarily to add more randomness. A better approach is to monitor diversity or improvement rate and apply exploration only when needed. Restarting stagnant whales, perturbing the best solution, maintaining subpopulations, or using entropy-based feedback can restore diversity without destroying useful information. This is more principled than applying a large mutation probability throughout the search.

If the weakness is weak local refinement, local search or exploitation-enhancing operators may be appropriate. Gaussian perturbation, pattern search, Nelder-Mead refinement, differential mutation around elite solutions, or gradient-informed steps can improve final accuracy. These methods are especially useful in continuous engineering design, where a good approximate solution must be refined to satisfy tight constraints. However, local search should be applied selectively because it can increase computational cost.

If the problem is constrained, the variant should include a constraint-handling mechanism from the beginning. Static penalties are easy to implement but may be sensitive to penalty coefficients. Adaptive penalties, feasibility rules, repair methods, decoders, and constraint-domination principles are often more reliable. Strong constrained WOA studies should report not only the objective value but also feasibility rate, maximum violation, average violation, and robustness across penalty settings.

If the problem is discrete or binary, the representation is the core contribution. A continuous update followed by arbitrary rounding may damage the search because small position differences may not correspond to meaningful solution changes. Transfer functions, probability vectors, permutation operators, swap sequences, or decoder-based representations should be selected according to the problem. For example, routing problems need permutation-preserving moves, while feature selection requires binary masks and sparsity-aware fitness.

If the problem is multi-objective, the design should include archive management, leader selection, and diversity preservation. A single global best solution is not suitable because the goal is a set of trade-off solutions. The archive should avoid becoming too large while preserving representative nondominated solutions. Grid density, crowding distance, reference vectors, or decomposition methods can guide leader selection and maintain diversity along the Pareto front.

If the objective evaluation is expensive, the variant should focus on evaluation efficiency rather than adding complex search equations. Surrogate modeling, parallel evaluation, adaptive sampling, early stopping, and transfer learning may provide larger benefits than another perturbation operator. The optimizer should also avoid unnecessary reevaluation of identical or infeasible solutions. Caching fitness values is a simple but often neglected technique.

Finally, every new WOA variant should include a complexity statement. Authors should state how many additional evaluations, memory structures, archive operations, model-training steps, or local-search calls are introduced. Without this information, readers cannot judge whether improved accuracy is worth the extra cost. Algorithmic quality is not only measured by solution value; it also includes efficiency, stability, simplicity, and reproducibility.

Table 9. Mechanism-driven design guidelines for developing stronger WOA variants.

Observed Problem	Recommended WOA Design Response	Evidence Needed for Validation
Low initial diversity	Use chaotic, opposition-based, quasi-random, or stratified initialization	Initial diversity metrics and equal evaluation-budget comparison
Premature convergence	Use adaptive exploration, restarts, entropy feedback, or multi-swarm search	Convergence curves, diversity curves, and stagnation analysis
Weak exploitation	Use local search, Gaussian perturbation, elite refinement, or gradient hints	Ablation showing improved final precision without excessive runtime
High dimensionality	Use decomposition, variable grouping, cooperative coevolution, or subspace search	Scaling tests across increasing dimensions
Discrete decisions	Use transfer functions, decoders, or problem-specific moves	Feasibility rate and comparison against discrete baselines
Constrained feasibility	Use repair, feasibility rules, adaptive penalties, or constraint domination	Objective values plus violation and feasibility statistics
Multiple objectives	Use Pareto archive, diversity preservation, and leader selection	Hypervolume, IGD, spread, and archive-quality plots
Expensive evaluations	Use surrogate models, caching, or parallel evaluation	Wall-clock time, evaluation count, and surrogate-error analysis

11 | Quality Assessment of Whale Optimization Algorithm Evidence

The strength of a WOA paper depends not only on the proposed algorithm but also on the quality of evidence used to support it. Evidence quality can be examined through four validity dimensions: internal validity, external validity, construct validity, and conclusion validity. Applying these dimensions to WOA research helps identify why some empirical claims are persuasive while others remain weak.

Internal validity concerns whether the observed improvement is actually caused by the proposed method. In WOA research, internal validity is threatened when several operators are added simultaneously without ablation analysis, when parameter tuning is inconsistent across algorithms, or when additional evaluations are not counted. To improve internal validity, each component should be tested separately and all algorithms should operate under equal evaluation budgets.

External validity concerns whether results generalize beyond the tested problems. A WOA variant evaluated only on a small set of low-dimensional benchmark functions may not generalize to high-dimensional, constrained, noisy, or discrete problems. External validity improves when studies include several benchmark categories, multiple dimensions, real-world case studies, and comparisons across different problem structures.

Construct validity concerns whether the metrics actually measure the intended performance. For example, classification accuracy alone may not be an adequate metric for feature selection because it ignores feature subset size and stability. Similarly, best objective value alone is not enough for constrained optimization because it may ignore feasibility. Construct validity improves when metrics are aligned with problem goals and domain requirements.

Conclusion validity concerns whether the statistical conclusions are justified. A small number of runs, missing variance measures, or inappropriate statistical tests can lead to unreliable claims. Since WOA is stochastic, repeated independent runs are mandatory. Statistical testing should be accompanied by effect sizes, because a statistically significant difference may still be too small to matter practically.

The quality of implementation is another major issue. Two implementations of the same WOA variant may differ in boundary control, random-number generation, fitness caching, initialization, or constraint treatment. These differences can change performance. Therefore, open-source code and detailed pseudocode are not optional extras; they are essential for cumulative scientific progress.

A useful quality assessment score for WOA papers could include items such as clear problem formulation, complete equations, reproducible parameter settings, fair baselines, sufficient runs, statistical testing, ablation study, runtime reporting, code availability, and domain-specific validation. Such a scoring system would help reviewers evaluate whether a proposed variant provides genuine evidence rather than only numerical tables.

Table 10. Validity threats and mitigation strategies for WOA-based research.

Validity Dimension	Typical threat in WOA Research	Recommended Mitigation
Internal validity	Improvement may be caused by hidden tuning, extra evaluations, or untested components	Use ablation studies, equal budgets, and transparent tuning
External validity	Results may not generalize beyond selected functions or datasets	Test across benchmark categories, dimensions, and real applications
Construct validity	Metrics may not reflect the true goal of the problem	Use domain-specific metrics and report feasibility, cost, and stability
Conclusion validity	Claims may rely on too few runs or weak statistical evidence	Use repeated runs, nonparametric tests, confidence intervals, and effect sizes
Implementation validity	Different coding details may change algorithm behavior	Release code or provide precise pseudocode and boundary-handling rules

12 | Domain-Specific Recommendations

12.1 | Recommendations for Engineering Design

Engineering design studies should treat constraints as first-class components of the problem. For each candidate design, the study should report whether the solution is feasible, how close it is to active constraints, and whether small perturbations violate feasibility. This is important because a design that narrowly satisfies constraints in simulation may be unsafe under manufacturing tolerances or modeling uncertainty.

For structural and mechanical design, hybrid WOA should be compared not only with other metaheuristics but also with established engineering optimization methods when applicable. Sensitivity analysis should be conducted for load cases, material properties, and constraint thresholds. A robust design is preferable to a marginally lighter design that fails under small uncertainty.

12.2 | Recommendations for Machine Learning Applications

In ML, WOA is often used as a wrapper optimizer for feature selection or hyperparameter tuning. These studies must avoid data leakage by separating training, validation, and testing. Nested cross-validation is recommended when the same dataset is used for both model selection and performance estimation. Otherwise, reported performance may reflect overfitting to the validation data rather than true generalization.

Feature-selection studies should also report stability. If a WOA variant selects very different feature subsets across runs with similar accuracy, the resulting model may be difficult to interpret. Stability measures, selected-feature frequency, and domain relevance of features are particularly important in biomedical and high-stakes applications.

12.3 | Recommendations for Energy and Smart Systems

Energy applications should evaluate WOA under realistic operating scenarios. For photovoltaic MPPT, this means rapidly changing irradiance, partial shading, sensor noise, and hardware constraints. For economic dispatch and optimal power flow, this means generator limits, ramp-rate constraints, prohibited zones, valve-point effects, emissions, and renewable uncertainty. A simplified model may be useful for initial validation, but practical claims require realistic constraints.

Smart grid, IoT, and edge-computing applications should report scalability and latency. An optimizer that produces high-quality schedules but requires excessive computation may not be suitable for real-time systems. PWOA, distributed subpopulations, and lightweight surrogate models may be useful, but their communication overhead must be measured.

12.4 | Recommendations for Biomedical and Security Applications

Biomedical WOA studies should combine numerical performance with interpretability. In diagnosis and gene selection, accuracy is important, but selected features should also be medically meaningful and reproducible. For image segmentation, metrics should be combined with expert inspection when possible. Ethical and clinical relevance should not be reduced to a single optimization score.

Cybersecurity applications should evaluate robustness against changing attack distributions. Intrusion-detection models optimized on one dataset may fail when attack types evolve. WOA-based feature selection or classifier tuning should therefore be tested across multiple datasets or temporally separated data when available. False-positive and false-negative costs should also be discussed because they have different operational consequences.

Table 11. Domain-specific recommendations for stronger WOA application papers.

Application Area	Most Important Methodological Requirement	Additional Reporting Recommendation
Engineering design	Feasibility and constraint robustness	Report active constraints, violation statistics, and sensitivity analysis
ML	Proper validation without data leakage	Report feature count, stability, and independent test performance
Energy systems	Realistic operating constraints and uncertainty	Evaluate dynamic scenarios and report reliability indicators
IoT/Cloud/Edge	Scalability and runtime overhead	Report latency, energy, makespan, and resource utilization
Biomedical analysis	Clinical or biological interpretability	Report domain-relevant metrics and expert validation where possible
Cybersecurity	Robustness to evolving threats	Report false positives, false negatives, and cross-dataset generalization

13 | Roadmap for the Next Generation of Whale Optimization Algorithm Research

The next generation of WOA research should be less focused on naming new variants and more focused on building reliable optimization frameworks. A short-term priority is reproducibility: common benchmark suites, reference implementations, and reporting standards would immediately improve the value of empirical comparisons. Such resources would also make it easier for researchers to identify genuinely useful modifications.

A medium-term priority is scalability. Many practical problems involve hundreds or thousands of variables, multiple constraints, and expensive evaluations. WOA variants should therefore incorporate decomposition, cooperative coevolution, adaptive sampling, and parallel evaluation. These mechanisms should be tested under increasing dimensionality and realistic computational budgets.

A long-term priority is theory-guided design. Metaheuristic research often advances empirically, but stronger theoretical analysis can help explain when and why an algorithm works. For WOA, theoretical work should examine diversity loss, attraction dynamics, boundary effects, convergence probabilities, and the impact of adaptive schedules. Even partial theory can improve design by identifying failure modes.

Another long-term direction is trustworthy optimization for high-stakes systems. When WOA is used in medical, power, security, or autonomous systems, performance must be reliable under uncertainty. This requires robust optimization, uncertainty quantification, explainability, and safety-aware constraints. WOA should not only find good solutions but also provide evidence that those solutions remain acceptable under perturbation.

The field should also develop stronger connections with automated algorithm design. Instead of manually combining operators, researchers can use algorithm configuration, reinforcement learning, or hyper-heuristics to select operators based on landscape features. However, automated design must be evaluated carefully to avoid overfitting to benchmark suites. Generalization across problem classes should become a central evaluation criterion.

Table 12. Proposed roadmap for future WOA research.

Time Horizon	Research Priority	Expected Contribution
Short term	Open-source reference implementations and reporting standards.	More reproducible and comparable WOA studies.
Short term	Ablation-based evaluation of existing hybrid variants.	Clearer understanding of which operators actually help.
Medium term	Large-scale, constrained, and DWOA frameworks.	Better applicability to realistic engineering systems.
Medium term	Surrogate-assisted and PWOA for expensive objectives.	Reduced computational cost and improved scalability.
Long term	Theory-guided analysis of convergence and diversity dynamics.	More principled design of adaptive mechanisms.
Long term	Trustworthy WOA for high-stakes domains.	Optimization methods that are robust, explainable, and safe.

14 | Conclusion

This expanded survey has presented a stronger and more critical review of the WOA. It has clarified the canonical model, corrected the mathematical presentation, explained the algorithm's strengths and weaknesses, and organized WOA variants according to the search components they modify. The survey shows that WOA remains attractive because of its simplicity, intuitive search behavior, and adaptability across many problem domains.

The analysis also shows that most successful WOA variants respond to specific limitations: premature convergence, loss of diversity, weak local search, poor discrete representation, insufficient constraint handling, expensive evaluation, or conflicting objectives. Adaptive schedules, chaotic maps, mutation strategies, hybrid operators, binary transformations, Pareto archives, surrogate models, and parallel subpopulations should therefore be understood as targeted design responses rather than arbitrary additions.

For application domains, WOA has demonstrated value in engineering design, power systems, ML, biomedical analysis, image processing, IoT, cloud/edge computing, cybersecurity, logistics, and transportation. Nevertheless, practical usefulness depends on formulation quality, feasibility handling, validation design, and computational cost. A WOA solution is strong only when it is both algorithmically justified and empirically validated under realistic conditions.

The paper recommends that future WOA research emphasize mechanism-driven design, rigorous benchmarking, statistical testing, ablation analysis, open-source implementation, and careful domain validation. Rather than multiplying variants without clear justification, the field should move toward theoretically informed, reproducible, scalable, and explainable WOA frameworks. Such a direction will strengthen the scientific contribution of WOA and increase its reliability in real-world optimization.

Authors' Contributions

All aspects of the research and manuscript preparation were carried out by the author. The author has read and approved the final version of the manuscript.

Funding

Not applicable.

Data Availability

All data supporting the reported findings in this research paper are provided within the manuscript.

Conflict of Interest

The author declares that they do not have any conflict of interest.

Consent for Publication

The author confirms consent for the publication of this work.

Ethics Approval and Consent to Participate

This article does not contain any studies with human participants performed by the author.

References

[1] Almufti, S. M. (2025). Comparative analysis of metaheuristic algorithms for solving the travelling salesman problems. *International Journal of Scientific World*, *11*(2), 26–30. https://doi.org/10.14419/7fk7k945

[2] Almufti, S. M., Shaban, A. A., Ali, Z. A., Ali, R. I., & Fuente, J. A. Dela. (2023). Overview of metaheuristic algorithms. *Polaris Global Journal of Scholarly Research and Trends*, *2*(2), 10–32. https://doi.org/10.58429/pgjsrt.v2n2a144

[3] Almufti, S. M., Marqas, R. B., Othman, P. S., & Sallow, A. B. (2021). Single-based and population-based metaheuristics for solving np-hard problems. *Iraqi Journal of Science*, *62*(5), 1710–1720. https://doi.org/10.24996/10.24996/ijs.2021.62.5.34

[4] Kaveh, A., & Bakhshpoori, T. (2019). *Metaheuristics: Outlines, MATLAB codes and examples*. Springer. https://doi.org/10.1007/978-3-030-04067-3

[5] Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, *95*, 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

[6] Nadimi Shahraki, M. H., Zamani, H., Asghari Varzaneh, Z., & Mirjalili, S. (2023). A systematic review of the whale optimization algorithm: Theoretical foundation, improvements, and hybridizations. *Archives of Computational Methods in Engineering*, *30*(7), 4113–4159. https://doi.org/10.1007/s11831-023-09928-7

[7] Rana, N., Latiff, M. S. A., Abdulhamid, S. M., & Chiroma, H. (2020). Whale optimization algorithm: A systematic review of contemporary applications, modifications and developments. *Neural Computing and Applications*, *32*(20), 16245–16277. https://doi.org/10.1007/s00521-020-04849-z

[8] Gharehchopogh, F. S., & Gholizadeh, H. (2019). A comprehensive survey: Whale optimization algorithm and its applications. *Swarm and Evolutionary Computation*, *48*, 1–24. https://doi.org/10.1016/j.swevo.2019.03.004

[9] Kirkpatrick, S., Gelatt Jr, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*(4598), 671–680. https://doi.org/10.1126/science.220.4598.671

[10] Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95-International Conference on Neural Networks* (Vol. 4, pp. 1942–1948). IEEE. https://doi.org/10.1109/ICNN.1995.488968

[11] Rashedi, E., Nezamabadi Pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences*, *179*(13), 2232–2248. https://doi.org/10.1016/j.ins.2009.03.004

[12] Rahimnejad, A., Akbari, E., Mirjalili, S., Gadsden, S. A., Trojovsky, P., & Trojovská, E. (2023). An improved hybrid whale optimization algorithm for global optimization and engineering design problems. *PeerJ Computer Science*, *9*, e1557. https://doi.org/10.7717/peerj-cs.1557

[13] Wei, J., Gu, Y., Lu, B., & Cheong, N. (2025). RWOA: A novel enhanced whale optimization algorithm with multi-strategy for numerical optimization and engineering design problems. *PloS One*, *20*(4), e0320913. https://doi.org/10.1371/journal.pone.0320913

[14] Kaur, G., & Arora, S. (2018). Chaotic whale optimization algorithm. *Journal of Computational Design and Engineering*, *5*(3), 275–284. https://doi.org/10.1016/j.jcde.2017.12.006

[15] Jiang, R., Yang, M., Wang, S., & Chao, T. (2020). An improved whale optimization algorithm with armed force program and strategic adjustment. *Applied Mathematical Modelling*, *81*, 603–623. https://doi.org/10.1016/j.apm.2020.01.002

[16] Yang, W., Xia, K., Fan, S., Wang, L., Li, T., Zhang, J., & Feng, Y. (2022). A multi-strategy whale optimization algorithm and its application. *Engineering Applications of Artificial Intelligence*, *108*, 104558. https://doi.org/10.1016/j.engappai.2021.104558

[17] Nadimi Shahraki, M. H., Zamani, H., & Mirjalili, S. (2022). Enhanced whale optimization algorithm for medical feature selection: A COVID-19 case study. *Computers in Biology and Medicine*, *148*, 105858. https://doi.org/10.1016/j.compbiomed.2022.105858

[18] Hussien, A. G., Hassanien, A. E., Houssein, E. H., Bhattacharyya, S., & Amin, M. (2018). S-shaped binary whale optimization algorithm for feature selection. In *Recent Trends in Signal and Image Processing: ISSIP 2017* (pp. 79–87). Springer. https://doi.org/10.1007/978-981-10-8863-6_9

[19] Alwajih, R., Abdulkadir, S. J., Al Hussian, H., Aziz, N., Al Tashi, Q., Mirjalili, S., & Alqushaibi, A. (2022). Hybrid binary whale with Harris Hawks for feature selection. *Neural Computing and Applications*, *34*(21), 19377–19395. https://doi.org/10.1007/s00521-022-07522-9

[20] Cao, H., Yan, B., Dong, L., & Yuan, X. (2024). Multipopulation whale optimization-based feature selection algorithm and its application in human fall detection using inertial measurement unit sensors. *Sensors*, *24*(24), 7879. https://doi.org/10.3390/s24247879

[21] Huang, M., Zhai, Q., Chen, Y., Feng, S., & Shu, F. (2021). Multi-objective whale optimization algorithm for computation offloading optimization in mobile edge computing. *Sensors*, *21*(8), 2628. https://doi.org/10.3390/s21082628

[22] Got, A., Moussaoui, A., & Zouache, D. (2020). A guided population archive whale optimization algorithm for solving multiobjective optimization problems. *Expert Systems with Applications*, *141*, 112972. https://doi.org/10.1016/j.eswa.2019.112972

[23] Makhadmeh, S. N., Kassaymeh, S., Rjoub, G., Bataineh, B., Sanjalawe, Y., & Al Betar, M. A. (2025). Recent advances in multi-objective whale optimization algorithm, its versions and applications. *Journal of King Saud University Computer and Information Sciences*, *37*(7), 200. https://doi.org/10.1007/s44443-025-00184-2

[24] Kudela, J., & Matousek, R. (2022). New benchmark functions for single-objective optimization based on a zigzag pattern. *IEEE Access*, *10*, 8262–8278. https://doi.org/10.1109/ACCESS.2022.3144067

Appendix

Appendix A. Practical checklist for designing a WOA study.

A practical WOA study should begin by defining the optimization problem with sufficient precision. The objective function, decision variables, constraints, bounds, and evaluation procedure must be stated clearly. If the problem is discrete or mixed-variable, the encoding and decoding scheme should be explained before any algorithmic result is presented. If the problem is multi-objective, the trade-off objectives and decision-maker preferences should be defined.

The next step is selecting an appropriate WOA variant. The selection should follow the problem structure. Continuous unconstrained problems may use canonical or adaptive WOA; high-dimensional problems may require decomposition; binary feature selection requires transfer functions; constrained engineering problems require repair or feasibility rules; and expensive simulations may require surrogate assistance. The variant should not be selected merely because it is recent or popular.

Experimental design should then establish fair comparison. Competing algorithms should receive equal evaluation budgets, and parameter settings should be obtained from original papers or tuned using a transparent protocol. Random seeds, independent runs, and hardware environment should be recorded. For machine-learning applications, the data split must prevent leakage between training, validation, and testing.

Finally, the results should be interpreted cautiously. Statistical significance should be combined with effect size and computational cost. If a variant improves accuracy by a very small margin while doubling runtime, the practical value may be limited. Strong conclusions should be restricted to the tested problem classes, and limitations should be acknowledged explicitly.

Table A1. Submission checklist for rigorous WOA research.

Checklist Item	Question to Answer Before Submission
Problem definition	Are objective functions, variables, constraints, and bounds fully specified?
Variant justification	Does each modification address a clearly identified weakness of WOA?
Baseline fairness	Are all algorithms compared under equal fitness-evaluation budgets?
Parameter reporting	Are population size, iterations, coefficients, and tuning method disclosed?
Statistical validity	Are enough independent runs and suitable statistical tests used?
Ablation	Is the contribution of each added operator tested separately?
Reproducibility	Are code, seeds, datasets, and experimental settings available or described?
Practical relevance	Are runtime, feasibility, scalability, and domain-specific metrics reported?

Appendix B. Reporting template for WOA experiments.

The following template is included to make the survey more directly useful for authors preparing WOA manuscripts. A WOA experiment should first state the optimization formulation in a compact mathematical form. This includes the objective function, decision vector, lower and upper bounds, equality and inequality constraints, dimensionality, and whether the problem is continuous, binary, integer, permutation-based, or mixed-variable. When the objective function is derived from simulation, the simulation settings should be documented so that the evaluation is reproducible.

The algorithm configuration should then be reported in a separate subsection. This subsection should include population size, maximum iteration count, total fitness-evaluation budget, initialization method, boundary-handling method, coefficient schedule, probability settings, mutation or local-search probabilities, archive size for multi-objective problems, and stopping conditions. If a parameter is tuned, the tuning range and selection criterion should be stated. If parameters are adopted from previous literature, the reference should be cited and any changes should be justified. The comparison protocol should be written before results are discussed. It should identify baseline algorithms, explain why they were selected, and state whether their parameters were taken from original papers, tuned by the authors, or set using a common protocol. All algorithms should be run under comparable computational budgets. When a hybrid WOA includes additional local search or opposition-based evaluations, these evaluations should be counted unless a clear reason is provided.

The result tables should include more than the best solution. Recommended columns include best, worst, mean, median, standard deviation, success rate, average runtime, and rank. For constrained problems, additional columns should show feasible rate and average constraint violation. For machine-learning problems, results should include accuracy or error plus model complexity, feature count, and validation protocol. For multi-objective problems, hypervolume and IGD should be accompanied by representative Pareto-front plots. The discussion section should interpret results rather than only repeating numerical rankings. Authors should explain why the proposed WOA variant performs well or poorly on specific functions or applications. For example, strong performance on multimodal functions may indicate effective exploration, while weak performance on rotated functions may indicate sensitivity to variable interaction. This interpretation is what transforms a numerical experiment into a scientific contribution.

Finally, limitations should be stated explicitly. A WOA variant may be strong on medium-dimensional continuous problems but untested on discrete or dynamic problems. It may produce accurate solutions but require high runtime. It may work well on benchmark functions but need additional validation on real data. Acknowledging such boundaries strengthens the manuscript because it shows that the authors understand the scope of their evidence.

Table B1. Recommended reporting template for WOA experimental manuscripts.

Manuscript Component	Minimum Information to Include
Problem formulation	Objective, variables, dimensionality, bounds, constraints, representation, and evaluation procedure.
Algorithm settings	Population size, iterations, evaluation budget, coefficient schedules, initialization, boundary handling, and stopping criteria.
Variant components	Exact description of each added mechanism and how it modifies canonical WOA.
Baselines	Algorithm names, parameter settings, references, tuning method, and equal-budget policy.
Statistical protocol	Number of runs, random seeds, summary statistics, tests, post-hoc analysis, and effect sizes.
Application validation	Domain-specific metrics, feasibility, runtime, scalability, and sensitivity analysis.
Reproducibility package	Code, datasets, seed files, benchmark definitions, and raw result tables when possible.

Appendix C. Operator-level interpretation of WOA variants.

Many WOA variants can be understood as operator-level changes to a small number of algorithmic functions. The initialization operator determines the first population. The leader operator determines which solution guides the movement. The movement operator determines how candidate positions change. The perturbation operator determines how diversity is restored. The selection operator determines which solutions survive. The constraint operator determines whether a candidate is feasible or repairable. The archive operator stores nondominated or elite solutions. Describing a variant in terms of these operators makes the method easier to compare and reproduce.

For example, a chaotic adaptive WOA may be represented as a canonical WOA in which the initialization operator uses a chaotic map and the control operator uses a nonlinear schedule. A hybrid WOA-DE may be represented as a canonical WOA in which some movement steps are replaced by differential mutation and crossover. A binary WOA may be represented as a canonical WOA with an added transfer operator that maps continuous position values into binary decisions. This operator-level description is clearer than simply naming the hybrid.

Operator-level interpretation also supports modular experimentation. Researchers can replace one operator at a time while keeping the rest of the algorithm fixed. This produces cleaner ablation studies and helps identify which operator is responsible for improvement. It also makes it possible to build reusable WOA libraries in which initialization, movement, perturbation, selection, and constraint handling are interchangeable modules.

A final advantage is conceptual transparency. When a paper says that a new WOA variant improves exploration, the reader should be able to identify which operator creates that improvement. If exploration improves because of a Lévy perturbation, this should be visible in the algorithm description. If exploitation improves because of local search, the frequency and cost of local search should be reported. Transparent operator descriptions reduce ambiguity and make WOA research easier to evaluate.

Table C1. Operator-level view for describing and comparing WOA variants.

Operator	Role in WOA Framework	Examples of Possible Modifications
Initialization	Generates the first population within the search bounds.	Chaotic maps, Latin hypercube sampling, opposition-based learning, orthogonal design.
Leader selection	Determines the solution or archive member that guides movement.	Elite selection, random elite choice, Pareto archive leader, crowding-based leader.
Movement	Updates whale positions using encircling, spiral, or exploration rules.	Nonlinear coefficients, PSO velocity, DE mutation, GWO hierarchy, SCA movement.
Perturbation	Restores diversity or escapes local optima.	Gaussian mutation, Cauchy mutation, Lévy flight, random walk, restart.
Representation	Maps candidate positions into valid decisions.	Binary transfer functions, permutation decoders, integer repair, mixed-variable encoding.
Constraint handling	Controls feasibility and violation treatment.	Static/adaptive penalty, repair, feasibility rules, constraint domination.
Archive/memory	Stores useful solutions for later selection.	External Pareto archive, elite memory, historical best population, surrogate training set.

Appendix D. Evaluation matrix for future WOA surveys.

Future surveys of WOA can be strengthened by using a structured evaluation matrix instead of summarizing papers only by year, author, and application. The matrix should record the modified component of the algorithm, the motivation for modification, the benchmark type, the dimensionality, the comparison baselines, the statistical tests, the application domain, and whether source code or datasets are available. Such a matrix allows readers to identify evidence gaps more quickly.

A survey matrix should also distinguish between methodological novelty and application novelty. A paper may be methodologically novel if it introduces a new adaptive schedule, a new leader-selection mechanism, or a new discrete representation. Another paper may be application-novel if it applies an existing WOA variant to a new engineering problem with a realistic formulation. Both kinds of contribution can be valuable, but they should not be evaluated using the same criteria.

The evidence level of each paper can be categorized qualitatively. A low-evidence paper may provide a new variant but test it on few functions without statistical analysis. A medium-evidence paper may include repeated runs and several baselines but lack ablation or code. A high-evidence paper should include diverse benchmarks, fair evaluation budgets, statistical testing, ablation, runtime analysis, and reproducible implementation. This classification would make future surveys more critical and less descriptive.

Another useful survey dimension is problem suitability. WOA is not equally suitable for every optimization task. Future reviews should record whether the problem is continuous, binary, integer, permutation-based, constrained, dynamic, noisy, expensive, multi-objective, or many-objective. This would help readers see which WOA mechanisms are most often used for each problem type and where research remains underdeveloped.

Finally, future surveys should examine negative or neutral results when available. Metaheuristic literature often emphasizes successful improvements, but unsuccessful or marginal results are scientifically informative because they reveal limits. A balanced survey should not only report where WOA performs well but also identify problem types where it struggles, conditions where simpler algorithms are competitive, and cases where added complexity does not justify the improvement.

Table D1. Evaluation matrix for future high-quality WOA surveys.

Evaluation Dimension	Recommended Coding Categories for Future Surveys
Modified WOA component	Initialization, control schedule, movement, perturbation, representation, archive, constraint handling, parallelization.
Problem type	Continuous, binary, integer, permutation, constrained, noisy, dynamic, expensive, multi-objective, many-objective.
Evidence level	Low: limited tests; medium: repeated tests and baselines; high: statistics, ablation, runtime, and reproducibility.
Benchmark coverage	Unimodal, multimodal, rotated, shifted, composition, engineering design, real-world application.
Validation method	Repeated independent runs, cross-validation, sensitivity analysis, feasibility analysis, Pareto-front indicators.
Reproducibility	Complete pseudocode, parameter settings, seeds, source code, datasets, and raw results.
Practical relevance	Runtime, scalability, interpretability, domain constraints, and deployment feasibility.

Appendix E. Glossary of key WOA terms.

This glossary clarifies recurring terms used throughout the survey. It is included to improve readability for researchers who are new to WOA while preserving technical precision for advanced readers. Consistent terminology is important because many WOA papers use similar words such as exploration, diversification, perturbation, mutation, and randomization without clearly separating their meanings.

In this paper, exploration refers to search behavior that increases the chance of discovering new promising regions, while exploitation refers to behavior that refines solutions near already promising regions. Diversity describes the spread of candidate solutions in the search space. Stagnation means that the population no longer produces meaningful fitness improvement. A leader is the solution or archive member that guides the position update. These concepts are useful for comparing WOA variants because they describe search behavior rather than algorithm names.

Table E1. Glossary of important terms used in WOA research.

Term	Meaning in WOA Research
Exploration	Global search behavior intended to discover new regions and reduce local-optimum trapping.
Exploitation	Local refinement around promising solutions, usually guided by the current best whale or archive leader.
Diversity	The degree to which population members occupy different regions of the search space.
Stagnation	A state in which repeated iterations produce little or no improvement in objective value.
Leader	The best solution or selected archive member used to guide the movement of other whales.
Transfer function	A mapping that converts continuous WOA positions into binary or discrete decisions.
Archive	A memory structure used to store elite or nondominated solutions, especially in multi-objective WOA.
Ablation study	An experiment that removes or isolates components to measure their individual contribution.

Using this terminology consistently improves the clarity of WOA studies. For instance, a method that changes the initial population should be described as an initialization or diversity mechanism, not simply as an improved optimizer. Likewise, a method that adds Gaussian disturbance near elite solutions should be interpreted primarily as a local refinement mechanism, whereas a method based on heavy-tailed jumps should be interpreted as an escape or diversification mechanism.

Clear terminology also supports fair comparison among variants. Two algorithms may have different names but perform similar functions if both modify the same operator. Conversely, two algorithms may share the same label, such as adaptive WOA, while using very different feedback rules. Describing the exact search role of each component reduces ambiguity and helps readers understand whether a contribution is genuinely new.

For authors, the glossary can be used as a checklist while writing the methodology section. Each proposed component should be connected to one or more of the listed terms, and the experiment should include evidence that the intended effect occurred. For example, if a component is claimed to improve diversity, the paper should report a diversity indicator or at least a convergence analysis that supports the claim.

Appendix F. Comprehensive catalogue of WOA modifications and hybrid variants.

This appendix integrates the extensive variant catalogue from the original manuscript with the mechanism-oriented taxonomy developed in the expanded survey. The aim is not to treat every named variant as equally validated, but to preserve the breadth of reported WOA modifications while clarifying that each one should be assessed by its search role, computational cost, and empirical evidence.

Table F1. Combined catalogue of WOA modifications and hybrid variants retained from the original manuscript and aligned with the expanded taxonomy.

Number	Modification / Hybrid Type	Description
1	Adaptive WOA (AWOA)	Dynamically adjusts control parameters to maintain balance between exploration and exploitation throughout iterations.
2	Improved WOA (IWOA)	Refines the encircling and spiral equations to improve local search efficiency.
3	Chaotic WOA (CWOA)	Employs chaotic maps for parameter updates to improve population diversity.
4	Inertia-Based WOA (ILWOA/MWOA)	Introduces inertia-like terms to prevent stagnation and accelerate convergence.

Table F1. Continued.

Number	Modification / Hybrid Type	Description
5	WOA–PSO Hybrid	Combines WOA's exploration and PSO's exploitation for balanced performance.
6	WOA–DE Hybrid	Integrates DE's mutation–crossover operators to enhance convergence speed.
7	WOA–GA Hybrid	Embeds genetic crossover and selection operators for improved diversity.
8	Binary WOA (BWOA)	Adapts WOA for discrete problems using binary transfer functions.
9	Multi-Objective WOA (MOWOA)	Extends WOA for Pareto-based multi-objective optimization.
10	Quantum WOA (QWOA)	Incorporates qubit representation and quantum superposition principles.
11	Levy Flight WOA (LWOA)	Uses Lévy flights for improved global exploration.
12	Opposition-Based WOA (OBWOA)	Applies opposition-based learning for faster and more accurate convergence.
13	Hybrid WOA–GWO	Combines WOA with GWO to balance exploration and exploitation.
14	Hybrid WOA–SCA	Integrates SCA dynamics to strengthen global search.
15	Chaos-Driven Adaptive WOA (CAWOA)	Combines chaotic initialization and adaptive parameter control.
16	Random Walk WOA (RWWOA)	Incorporates stochastic random walks for enhanced exploration.
17	Elite Opposition-Based WOA (EOBWOA)	Integrates elite opposition learning for faster convergence.
18	Fuzzy WOA (FWOA)	Uses fuzzy logic to adjust parameters adaptively based on search performance.
19	MBWOA	Introduces Gaussian or Cauchy mutations to diversify the population.
20	Ensemble WOA (EWOA)	Combines multiple control strategies or sub-algorithms concurrently.
21	PWOA	Distributes computation across multiple nodes for faster convergence.
22	DL–WOA	Uses WOA as an optimizer for neural network training and hyperparameter tuning.
23	Hybrid WOA–SA	Combines simulated annealing's probabilistic acceptance with WOA's search.
24	Hybrid WOA–ACO	Incorporates pheromone-guided path construction for routing and scheduling.
25	DWOA	Employs time-varying control coefficients for adaptive exploration intensity.
26	Orthogonal WOA (OWOA)	Uses orthogonal experimental design for efficient initialization.
27	Enhanced WOA (EWOA)	General category for improved exploitation or parameter re-scaling variants.
28	Self-Adaptive WOA (SAWOA)	Self-regulates control parameters using feedback from fitness evolution.
29	Improved Convergence WOA (ICWOA)	Implements nonlinear convergence control for faster optimization.
30	Gaussian WOA (GWOA)	Adds Gaussian perturbation to intensify local refinement.

Table F1. Continued.

Number	Modification / Hybrid Type	Description
31	Cauchy WOA (CWOA2)	Employs Cauchy distribution mutation to improve search diversity.
32	Exponential Decay WOA (EDWOA)	Introduces exponential decay control on parameters for smoother convergence.
33	Differential WOA (DWOA2)	Integrates differential update mechanisms for enhanced diversity.
34	Entropy-Controlled WOA (EnWOA)	Utilizes information entropy to balance search adaptively.
35	Gradient-Guided WOA (GGWOA)	Uses gradient direction information for faster convergence.
36	Momentum-Driven WOA (M-WOA)	Applies momentum to reduce oscillation in position updates.
37	K-Means Assisted WOA (KM–WOA)	Employs clustering to maintain diversity around promising regions.
38	Hybrid WOA–BAT	Combines echolocation-inspired BAT algorithm with WOA's mechanisms.
39	Hybrid WOA–Firefly (WOA–FA)	Uses light-intensity-based attraction to improve exploration.
40	Hybrid WOA–ABC	Merges foraging-based search from ABC to refine local search.
41	Hybrid WOA–HS	Incorporates harmony memory and pitch adjustment mechanisms.
42	Hybrid WOA–TLBO	Leverages teaching–learning phases to strengthen WOA's learning.
43	Hybrid WOA–CSA	Combines crow memory strategy with WOA's encircling behavior.
44	Hybrid WOA–MFO	Integrates Moth Flame Optimization's flight path search pattern.
45	Hybrid WOA–SMA	Employs slime mould oscillatory behavior to enhance adaptability.
46	Hybrid WOA–ALO	Uses trapping and random walk of ALO for exploratory improvement.
47	Hybrid WOA–SOA	Combines seagull migration and attack phases with WOA's model.
48	Hybrid WOA–CS	Embeds Lévy-flight-based cuckoo dynamics for random search.
49	Hybrid WOA–MPA	Fuses marine predator–prey dynamics with WOA's exploitation.
50	Hybrid WOA–EPO	Incorporates penguin grouping and thermal dynamics principles.
51	Hybrid WOA–GJO	Combines golden jackal chasing strategy with WOA's bubble-net phase.
52	Multi-Population WOA (MPWOA)	Divides population into multiple sub-swarms for diversity maintenance.
53	Multi-Swarm Cooperative WOA (MSWOA)	Enables information sharing among multiple WOA subgroups.
54	Ensemble Learning–Driven WOA (EL–WOA)	Uses ensemble decision mechanisms among WOA variants.
55	SA–WOA	Employs surrogate modeling to handle computationally expensive evaluations.

Table F1. Continued.

Number	Modification / Hybrid Type	Description
56	Transfer Learning–Based WOA (TL–WOA)	Transfers prior optimization knowledge to new problem instances.
57	Reinforcement Learning–WOA (RL–WOA)	Utilizes RL agents to dynamically adjust parameters and strategies.
58	Dynamic Population WOA (DPWOA)	Adjusts population size based on diversity and convergence indicators.
59	Energy-Balanced WOA (EBWOA)	Applies bio-inspired energy consumption model to guide search.
60	Thermal Exchange WOA (TEWOA)	Adapts heat transfer theory for dynamic search adaptation.
61	Modified Spiral Search WOA (SSWOA)	Introduces variable-radius spiral equations for refined exploitation.
62	Neighborhood Search WOA (NSWOA)	Incorporates local neighbor interactions for improved convergence.
63	Orthogonal Chaotic WOA (OCWOA)	Combines orthogonal learning and chaos for initialization.
64	Improved Discrete WOA (IDWOA)	Employs customized mapping for integer and permutation optimization.
65	Hybrid WOA–EO	Integrates equilibrium optimizer's exploration rules.
66	Hybrid WOA–SFO	Uses sailfish-inspired attacking–migrating strategies.
67	Improved Multi-Objective WOA (IMOWOA)	Incorporates fuzzy dominance and crowding distance ranking.
68	Hybrid WOA–Jellyfish Search (WOA–JS)	Combines time-controlled motion from JS for adaptive exploration.

The catalogue shows that WOA research has expanded from simple parameter adaptation and chaotic randomization toward hybrid, multi-population, surrogate-assisted, and learning-guided search. However, future studies should avoid presenting a new label as sufficient evidence of novelty; the mechanism, ablation evidence, and suitability for a specific problem class must be made explicit.

Appendix G. Extended application catalogue of WOA.

This appendix preserves the broad application inventory from the original manuscript and links it to the stronger discussion of evaluation criteria in the expanded version. The table is useful as a practical index for researchers selecting a WOA variant for a new domain.

Table G1. Combined application catalogue of WOA across scientific, engineering, and intelligent-system domains.

Number	Application Domain	Specific Application
1	Structural optimization	Truss structure design, welded beam design, pressure vessel optimization, and tension/compression spring design
2	Energy systems	Photovoltaic MPPT, renewable energy integration, economic load dispatch, and microgrid operation
3	Power systems	Optimal power flow, generator scheduling, capacitor placement, and load forecasting
4	Image processing	Image segmentation, contrast enhancement, thresholding, and image reconstruction
5	Medical diagnosis	Liver segmentation, brain tumor detection, toxicity classification, and COVID-19 diagnosis
6	Data clustering and feature selection	High-dimensional feature selection, clustering, and dimensionality reduction

Table G1. Continued.

Number	Application Domain	Specific Application
7	Control systems	PID parameter tuning, adaptive control, and load frequency control
8	WSNs	Node deployment, energy-efficient clustering, and lifetime maximization
9	ML and DL	Hyperparameter tuning for SVM, CNN, and hybrid WOA–DL frameworks
10	Manufacturing and scheduling	Job-shop scheduling, process planning, and production sequencing
11	Communication networks	Resource allocation, channel estimation, and QoS optimization in 5G
12	Supply chain and logistics	Inventory optimization, routing, and transport cost minimization
13	Robotics and path planning	Path optimization and swarm-based multi-robot coordination
14	Financial systems	Portfolio optimization, risk management, and stock trend prediction
15	Water resources engineering	Reservoir operation and water distribution optimization
16	Transportation systems	Traffic control, route optimization, and intelligent scheduling
17	Cybersecurity and cryptography	Image encryption, key generation, and network intrusion detection
18	IoT	Energy-efficient clustering, resource optimization, and device deployment
19	Cloud and edge computing	Task allocation, VM scheduling, and resource provisioning
20	Bioinformatics	Gene selection, protein folding, and DNA sequence alignment
21	Renewable energy forecasting	Wind and solar power prediction using hybrid WOA–ML models
22	Structural health monitoring	Damage detection and modal parameter estimation
23	Environmental systems	Pollution control, waste management, and ecosystem modeling
24	Network security	IDS optimization, DoS prevention, and secure communication
25	Signal processing	Filter design, spectral estimation, and biomedical signal analysis
26	Software engineering	Test case prioritization and software effort estimation
27	Hydrology	Flood forecasting and watershed modeling
28	Aerospace engineering	UAV path planning and aerodynamic shape optimization
29	Smart grids	Load management, power prediction, and demand response
30	Combinatorial optimization	TSP, knapsack, and assignment problems

The application catalogue confirms that WOA is most defensible when the target problem is nonlinear, derivative-free, multimodal, constrained, discrete, or expensive to evaluate. For deployment-oriented research, the algorithm should be assessed not only by best objective value but also by feasibility, runtime, robustness, and reproducibility.