Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Atmospheric Turbulence-Driven Chaotic Cloud Drift Optimization for Feature Selection and Forecasting

Abdelmonem M. Ibrahim
Al-Azhar University
Doaa A. Fakhry
doaaahmedfakhry@gmail.com
Assiut University
Ahmed Abdel-Monsef Allam
Assiut University

Research Article

Keywords: Biomedical datasets, Chaotic Maps, Cloud Drift Optimization, Feature Selection, Forecasting, Classification, Machine Learning

Posted Date: July 30th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10373695/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Atmospheric Turbulence-Driven Chaotic Cloud Drift Optimization for Feature Selection and Forecasting

Abdelmonem M. Ibrahim[1], Doaa A. Fakhry[2,3*], Ahmed Abdel-monsef Allam[2]

[1*]Department of Mathematics, Faculty of Science, Al-Azhar University (Assiut Branch), Assiut, Egypt.
[2*]Department of Mathematics (Computer Science Branch), Faculty of Science, Assiut University, Assiut, Egypt.
[3*]Department of Information Technology, Faculty of Industry and Energy Technology, New Assiut Technological University (N.A.T.U), Assiut, Egypt.

*Corresponding author(s). E-mail(s): doaaahmedfakhry@gmail.com;
Contributing authors: abdelmonem@azhar.edu.eg;
dallam51ahmed@yahoo.com;
†These authors contributed equally to this work.

Abstract

This paper introduces a novel Chaotic Improved Cloud Drift Optimization (CI-CDO) algorithm that enhances the original Cloud Drift Optimization (CDO) through a physically grounded atmospheric model driven by chaotic dynamics. The proposed framework redefines the search process using three complementary mechanisms: a *Cloud Splitting Mechanism* for global exploration, a *Cloud Merging Mechanism* for local exploitation, and an adaptive *Atmospheric Turbulence* mechanism governed by the Atmospheric Turbulence Index (***np***) to dynamically regulate the transition between them. Furthermore, a logistic chaotic map, adaptive drift coefficients, and a structured multi-candidate update strategy improve population diversity, strengthen convergence behavior, and reduce premature stagnation in complex optimization landscapes.

To evaluate its effectiveness, CI-CDO was comprehensively assessed through three experimental scenarios. First, it was benchmarked on the CEC 2021 test suite, where it achieved superior or highly competitive performance across

unimodal, multimodal, hybrid, and composition functions compared with state-of-the-art metaheuristic algorithms. Second, CI-CDO was adapted for binary feature selection on 21 UCI datasets using a wrapper-based k-Nearest Neighbors classifier, consistently achieving high classification accuracy while selecting compact feature subsets. Finally, the proposed algorithm was applied to real-world seasonal water quality and temperature forecasting using data collected from Egyptian monitoring stations, demonstrating strong predictive performance, robustness, and generalization capability.
Overall, the proposed atmospheric mechanisms and chaos-driven search strategy establish CI-CDO as a robust and physically inspired optimization framework for solving both benchmark optimization problems and real-world engineering and data science applications.

Keywords: Biomedical datasets, Chaotic Maps, Cloud Drift Optimization, Feature Selection, Forecasting, Classification, Machine Learning

1 Introduction

Real-world applications in data science and engineering frequently encounter complex challenges such as the curse of dimensionality, noisy measurements, and nonlinear system dynamics. These factors collectively hinder the accuracy, scalability, and efficiency of computational models [1]. In high-dimensional datasets, the presence of redundant and irrelevant features often degrades predictive performance and increases computational overhead, thereby emphasizing the importance of effective feature selection strategies [2]. Wrapper-based feature selection has gained prominence due to its direct evaluation of candidate subsets via predictive models, offering higher reliability in real-world scenarios [3, 4]. Among these models, the k-Nearest Neighbors (KNN) classifier remains a widely used choice due to its simplicity, robustness, and strong generalization capabilities across domains [5, 6]. For instance, biomedical datasets—such as those involving gene expression or epigenetic markers—often consist of thousands of attributes, necessitating robust selection strategies to isolate relevant biomarkers for disease diagnosis [6]. Similarly, forecasting problems in domains like energy, finance, and environmental science must address challenges related to temporal dependencies, non-stationarity, and uncertainty. In water resource management, predicting water quality indicators is essential for sustainable operations, yet the datasets involved are typically highly nonlinear and exhibit seasonal fluctuations [7]. In financial markets, the stochastic and volatile nature of time series data limits the effectiveness of traditional statistical approaches, prompting the adoption of adaptive, metaheuristic-based forecasting models [8]. The COVID-19 pandemic further revealed the shortcomings of conventional epidemiological models, highlighting the need for hybrid, optimization-driven predictors that can better capture complex transmission dynamics under uncertainty [9, 10]. To address these multifaceted issues, researchers have increasingly turned to metaheuristic optimization algorithms, which offer flexible and scalable frameworks for exploring complex search spaces. In feature selection, swarm intelligence and evolutionary computation have shown promise

in balancing classification accuracy and dimensionality reduction, leading to interpretable and computationally efficient models [5, 6]. Similarly, metaheuristic-driven parameter tuning has enhanced the adaptability and accuracy of forecasting models across various domains [7, 11]. Despite these advances, many existing algorithms still suffer from drawbacks such as premature convergence, insufficient exploration, and limited robustness when faced with highly dynamic or large-scale problems. These limitations have motivated the development of enhanced metaheuristic variants, particularly those incorporating chaotic systems, hybridization techniques, or multi-strategy frameworks that improve search diversity and performance consistency across diverse real-world scenarios [12–14]. Forecasting remains a critical task in domains such as energy, finance, healthcare, and environmental management, where accurate predictions directly influence decision-making and resource allocation. Time-series data, including seasonal variations in environmental indicators like water quality or temperature, often exhibit challenges such as non-stationarity, noise, and nonlinear dynamics that limit the effectiveness of traditional statistical models [15, 16]. To overcome these limitations, metaheuristic-driven parameter optimization has emerged as a powerful approach, enhancing the adaptability and robustness of predictive models under uncertainty [7, 8, 17]. By dynamically tuning model parameters, metaheuristics can significantly improve forecasting accuracy across diverse application areas, ranging from smart city infrastructure to medical prognosis and financial market prediction. Nevertheless, despite their promise, many existing optimization algorithms still suffer from drawbacks such as premature convergence, sensitivity to parameter settings, and limited scalability when applied to high-dimensional or highly nonlinear problems. These challenges have motivated the exploration of advanced algorithmic designs, including chaotic variants of metaheuristics, which introduce nonlinearity and ergodicity to enhance the balance between exploration and exploitation, thereby offering more reliable solutions in complex forecasting scenarios [18, 19]. Metaheuristics have become indispensable tools for solving real-world optimization problems such as feature selection and time-series forecasting, owing to their flexibility and robustness in handling complex and nonlinear search spaces. Early algorithms such as Genetic Algorithms (GA) [20, 21], Particle Swarm Optimization (PSO) [22, 23], and Differential Evolution (DE) [24, 25] laid the foundation for modern population-based optimization by balancing exploration (diversification) and exploitation (intensification). These approaches demonstrated effectiveness in selecting informative features from high-dimensional datasets and tuning predictive models for time-series analysis. Subsequent developments introduced Ant Colony Optimization (ACO) [26], Artificial Bee Colony (ABC) [27], and Gravitational Search Algorithm (GSA) [28], which offered enhanced exploration strategies and were successfully applied to biomedical feature selection and financial forecasting. More recent algorithms such as Grey Wolf Optimizer (GWO) [29, 30], Whale Optimization Algorithm (WOA) [31, 32], Harris Hawks Optimization (HHO) [33], and Equilibrium Optimizer (EO) [34] have achieved state-of-the-art results in both domains, particularly in handling high-dimensional feature spaces and highly nonlinear forecasting tasks. To further address challenges such as premature convergence, scalability, and sensitivity to parameter settings, researchers

have developed hybrid and chaotic variants of these algorithms, which exploit ergodicity and nonlinear dynamics to strengthen global exploration while maintaining strong local exploitation [6, 8]. Hybrid and parallel variants further enhance performance and scalability [35]. These innovations have significantly advanced the applicability of metaheuristics in solving challenging feature selection and forecasting problems across healthcare, energy, finance, and environmental systems. Chaotic metaheuristics integrate principles of chaos theory into optimization frameworks, leveraging deterministic yet unpredictable dynamics to enhance exploration, diversify the population, and reduce the risk of premature convergence. By embedding chaotic maps such as logistic, tent, or sine maps into the initialization or parameter control stages, these algorithms can achieve more effective coverage of the search space compared to their classical counterparts. For example, chaotic versions of Particle Swarm Optimization (CPSO) and Differential Evolution (CDE) have demonstrated faster convergence and improved accuracy in forecasting and engineering design problems [36, 37]. Similarly, chaotic Grey Wolf Optimizer (CGWO) and chaotic Whale Optimization Algorithm (CWOA) have shown superior performance in high-dimensional feature selection, maintaining a better balance between exploration and exploitation [38, 39]. More recently, chaos-enhanced Equilibrium Optimizer (CEEO) and chaotic Reptile Search Algorithm (CRSA) have yielded competitive results in biomedical feature selection, image recognition, and time-series forecasting, underscoring the growing role of chaos theory in tackling complex, nonlinear optimization problems [6, 40, 41]. Overall, the integration of chaotic dynamics strengthens the adaptability and robustness of metaheuristics, making them highly suitable for challenging tasks such as feature selection and predictive modeling under uncertainty. The Cloud Drift Optimization (CDO) algorithm is a recent nature-inspired metaheuristic that models cloud behavior in atmospheric systems. It simulates exploration through particle drift and exploitation through convergence to promising regions in the search space. While original and effective, CDO and its variants occasionally suffer from premature convergence and limited performance in dynamic or highly multimodal environments [42]. To overcome the limitations of conventional Cloud Drift Optimization (CDO) and its variants, this paper proposes the Chaotic Improved Cloud Drift Optimization (CI-CDO) algorithm. The key novelty of CI-CDO lies in the integration of chaotic dynamics at multiple stages of the search process. Specifically, chaotic maps are employed for population initialization, ensuring a more diverse and well-distributed set of candidate solutions at the start of the optimization. Moreover, adaptive drift coefficients guided by chaotic sequences are introduced to dynamically regulate the balance between exploration and exploitation, thereby reducing the risk of premature convergence. In addition, a refined multi-candidate update strategy is designed to enhance convergence behavior and improve solution quality by incorporating multiple guiding solutions rather than relying solely on the global best. These mechanisms collectively allow to adapt more flexibly to complex, nonlinear search landscapes. The effectiveness of CI-CDO is evaluated through three comprehensive experimental scenarios. First, the proposed algorithm is benchmarked on the 50 functions of the CEC 2021 test suite, including unimodal, multimodal, hybrid, and composition functions designed to assess the robustness, convergence behavior, and scalability of optimization algorithms. CI-CDO

is compared with several state-of-the-art metaheuristics and consistently achieves competitive or superior performance across the benchmark suite. Second, CI-CDO is extended to binary feature selection and evaluated on 21 benchmark datasets from the UCI repository within a wrapper-based framework using the k-Nearest Neighbors (KNN) classifier. The proposed algorithm effectively identifies compact subsets of informative features while maintaining or improving classification accuracy compared with recent binary optimization methods. Finally, CI-CDDO is applied to real-world seasonal water quality and temperature forecasting using data collected from Egyptian monitoring stations, where it demonstrates strong predictive performance, robustness, and generalization capability.

In summary, this work advances the Cloud Drift Optimization paradigm by introducing a physically inspired atmospheric evolution model that reformulates the search process through three complementary mechanisms: *Cloud Splitting* for global exploration, *Cloud Merging* for local exploitation, and an adaptive *Atmospheric Turbulence* mechanism that dynamically regulates the transition between them. Furthermore, the integration of logistic chaotic dynamics, adaptive drift coefficients, and a structured multi-candidate update strategy enhances population diversity, strengthens convergence stability, and effectively mitigates premature convergence. Extensive experimental results demonstrate that the proposed framework is both theoretically sound and practically effective across benchmark optimization, binary feature selection, and real-world forecasting applications, highlighting its potential as a robust general-purpose metaheuristic optimizer. The remainder of this article is organized as follows: Section 2 reviews literature on chaos-enhanced metaheuristics. Section 3 provides an overview of the original CDO algorithm and the chaotic maps used to enhance CDO. Section 4 defines the CI-CDO algorithm. Section 5 presents experiments and results. Section 10 concludes and suggests future directions.

2 Related Work

This section reviews the current state of chaotic metaheuristic algorithms, focusing on their applications in optimization, feature selection, and forecasting. It highlights how chaos theory has been integrated to improve exploration and convergence, positioning the proposed Chaotic Improved Cloud Drift Optimization (CI-CDO) within this landscape.

2.1 Chaotic Metaheuristics in Optimization

The integration of chaotic maps into optimization algorithms has attracted significant interest due to the improved balance between exploration and exploitation it offers. These chaos-augmented algorithms tend to avoid premature convergence and better explore complex solution spaces.

1. Chaotic Particle Swarm Optimization (CPSO): Particle Swarm Optimization (PSO), originally proposed by Kennedy and Eberhart [43], has been widely improved through chaotic enhancements. CPSO introduces chaotic sequences (e.g., Logistic or Tent maps) in velocity or position updates to increase swarm diversity

and prevent stagnation. CPSO has demonstrated superior performance on various benchmark problems, including training neural networks and complex function optimization [44, 45].

2. Chaotic Grey Wolf Optimizer (CGWO): Inspired by grey wolves' social hierarchy and hunting strategies [29], GWO has gained popularity for its simplicity and effectiveness. Chaotic variants, which integrate chaos in position updating or parameter tuning, have been shown to enhance the algorithm's global search capabilities and convergence speed in complex engineering problems [46, 47].
3. Chaotic Whale Optimization Algorithm (CWOA): The Whale Optimization Algorithm (WOA), which mimics humpback whale hunting behavior [31], benefits from chaotic maps in the encircling and bubble-net attacking phases. These enhancements help in preventing local optima trapping and improve convergence [48].
4. Chaotic Marine Predators Algorithm (CMPA): MPA, inspired by marine predator foraging strategies [49], has been improved through chaos to better explore the search space and escape local minima. It has seen successful applications in benchmark optimization [50].
5. Chaotic Sand Cat Swarm Optimization (SCSO): SCSO is inspired by sand cats' predatory behavior [51]. With chaotic initialization and update schemes, chaotic SCSO variants demonstrate improved convergence rates and solution quality [52].
6. Chaotic Heap-Based Optimizer (HBO): HBO uses a heap structure to manage candidate solutions [53]. Chaotic search operators in HBO help diversify search directions and improve global exploration [54].
7. Chaotic Cuckoo Search Algorithm (CCSA): Based on cuckoo brood parasitism [55], the CSA benefits from chaotic maps that improve exploration by dynamically perturbing nest positions, leading to superior performance on complex optimization and feature selection tasks [56].
8. Chaotic Electromagnetic Field Optimization (CEFO): Introduced the Electromagnetic Field Optimization algorithm [57], which integrates chaotic dynamics into the electromagnetic field optimization paradigm. By leveraging chaotic maps to guide the movement of charged particles, CEFO improved exploration capability, reduced the likelihood of premature convergence, and achieved superior performance on a wide range of benchmark functions.

2.2 Chaotic Metaheuristics in Feature Selection

Feature selection reduces data dimensionality and improves model performance. Chaotic metaheuristics have been widely applied due to their enhanced exploration abilities:

1. Chaotic PSO for Feature Selection: Chaotic PSO uses chaotic sequences to diversify initial populations and update velocities, which improves escape from local optima and helps select optimal feature subsets [58, 59].
2. Chaotic Ant Lion Optimizer (CALO): CALO integrates chaotic maps with the Ant Lion Optimizer (ALO), improving its ability to balance exploration and exploitation in feature subset selection [60].

3. Chaotic Selfish Herd Optimizer (CSHO): CSHO models group behavior under threat, where chaos helps avoid premature convergence and enhance exploration [61].
4. Chaotic Competitive Swarm Optimizer (ACCSO): ACCSO incorporates asymmetric chaotic maps to enhance competitive swarm dynamics in high-dimensional datasets [62].
5. Chaotic Butterfly Optimization Algorithm (CBOA): CBOA combines chaotic maps with butterfly-inspired movements to increase feature subset diversity [63].
6. Chaotic Firefly Algorithm (CFA): CFA uses chaotic sequences to perturb firefly attractiveness, enhancing feature selection performance [64].
7. Chaotic Grey Wolf Optimizer (CGWO): CGWO has also been applied to feature selection, where chaos improves population diversity and convergence [65].

2.3 Chaotic Metaheuristics in Forecasting

Forecasting real-world time series benefits from the enhanced parameter optimization offered by chaotic metaheuristics:

1. CPSO for Time Series Forecasting: CPSO has been applied to optimize forecasting models such as ANNs and SVMs, improving prediction accuracy and preventing premature convergence in parameter tuning [66, 67].
2. CGWO for Short-Term Load Forecasting: CGWO has been shown to enhance electrical load prediction accuracy by better exploring parameter spaces [68].
3. CWOA for Rainfall Prediction: CWOA effectively optimizes rainfall prediction model parameters, outperforming standard techniques [69].
4. CMPA for Wind Speed Forecasting: CMPA's chaotic mechanisms improve wind speed forecasting accuracy by enhancing exploration and avoiding local optima [70].
5. Chaotic SCSO for Temperature Forecasting: Chaotic SCSO has been effectively used for temperature forecasting, providing better parameter tuning and accuracy [71].

As shown, chaotic metaheuristics have emerged as a powerful extension of classical population-based optimization algorithms, exploiting the deterministic yet unpredictable dynamics of chaos to enhance exploration and prevent premature convergence.

3 Overview

3.1 The Cloud Drift Optimization (CDO) Algorithm

Cloud Drift Optimization (CDO) is a recently proposed nature-inspired optimization algorithm that mathematically models the dynamic behavior of clouds in the Earth's atmosphere [42]. Unlike conventional metaheuristics, CDO uniquely leverages two natural phases of cloud movement exploration and exploitation to efficiently navigate the search space and converge toward an optimal solution. The algorithm is based on the premise that clouds naturally gravitate toward areas with favorable atmospheric conditions (i.e., higher moisture content), which provides a natural target-seeking behavior. The algorithm's design is comprehensive and includes several key steps,

such as random initialization, dynamic weight adaptation, bimodal position updating, adaptive random perturbation, and an intelligent convergence condition. Each search agent, or "artificial cloud", is assigned a dynamic weight that influences its movement, mimicking the effect of heavier, more active clouds on their surroundings.

3.1.1 Problem Definition

The objective of CDO is to find a solution $\mathbf{x}^*$ that minimizes (or maximizes) an objective function $f(\mathbf{x})$ over a bounded search space Ω. The problem is defined as:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\Omega} f(\mathbf{x}) \tag{1}$$

where $\mathbf{x} = [x(1), x(2), \ldots, x(D)]$, and the bounds for each dimension are defined as $lb(j) \leq x(j) \leq ub(j)$.

3.1.2 Initialization

The algorithm begins by initializing a population of N candidate solutions (clouds) randomly within the search space. This ensures an unbiased and broad initial exploration, similar to the random distribution of cloud particles in the atmosphere.

$$X(i,j) = lb(j) + \text{rand}(0,1) \times (ub(j) - lb(j)) \tag{2}$$

Here, $X(i,j)$ represents the j-th dimension of the i-th cloud, and $\text{rand}(0,1)$ is a uniform random number in the range [0, 1].

3.1.3 Weight Adaptation

Each cloud is assigned a dynamic weight that is updated based on its fitness value relative to the rest of the population. This weight, $w(i,j)$, determines the influence of a particle's position on its own and others' movements. The weight is calculated using a fitness-proportional formula:

$$w(i,j) = \begin{cases} 1 + (0.3 + 0.7 \times \text{rand}(0,1)) \times \log_{10}\left(\frac{\text{Best_fitness} - f(i)}{S} + 1\right) & \text{if } f(i) \leq f_{\text{median}} \\ 1 - (0.3 + 0.7 \times \text{rand}(0,1)) \times \log_{10}\left(\frac{f(i) - \text{Best_fitness}}{S} + 1\right) & \text{if } f(i) > f_{\text{median}} \end{cases} \tag{3}$$

where Best_fitness is the best fitness value found so far, and $S = \text{Best_fitness} - \text{Worst_fitness} + \epsilon$. The values 0.3 and 0.7 are empirical constants that control the scaling of the weight.

3.1.4 Position Update

The position of each cloud is updated iteratively using a bimodal search strategy that includes distinct exploitation and exploration phases, see Algorithm 1.

Exploitation Phase: Local Refinement:

In this phase, clouds move toward the best-known solution, refining their positions. This behavior models how clouds adjust their formations to optimal atmospheric conditions. The update equation is:

$$X(i,j) = X_{best}(j) + 0.8 \cdot v_b(j) \cdot (w(i,j) \cdot X(A,j) - X(B,j)) \tag{4}$$

where v_b is a uniform random adjustment factor in the range $[-0.2a, 0.2a]$, and A and B are indices of randomly chosen clouds. The parameter $a = \text{atanh}(-t/T + 1)$ decreases over time, transitioning the algorithm toward more localized search.

Cloud Splitting Mechanism: Global Search:

This phase promotes population diversity and prevents stagnation by allowing clouds to move randomly within the search space. This mimics the unpredictable drift of clouds in the atmosphere.

$$X(i,j) = v_c(j) \cdot X(i,j), \tag{5}$$

where v_c is a uniform random factor in the range $[-0.2b, 0.2b]$, and $b = 1 - t/T$ is a parameter that decreases over time. The choice between the exploitation and exploration phases is determined by a random number compared to a probability value p (Eq. (6)).

$$p = \tanh(|f(i) - \text{Best_fitness}|), \tag{6}$$

where f_i denotes the fitness of the current solution and Best_fitness is the best-so-far fitness value.

Random Perturbations:

To further prevent premature convergence, a portion of the cloud population is randomly reinitialized with a probability z, which decreases as the algorithm progresses.

$$X(i,:) = (ub - lb) \cdot \text{rand}(1,d) + lb \quad \text{with probability } z, \tag{7}$$

where $z = 0.002 + 0.003 \cdot (1 - t/T)$ placed as probability to reduce random jumps over iterations. This behavior mimics sudden, unpredictable weather changes that can redirect a cloud's course, ensuring continued exploration even in later stages of the optimization. The algorithm terminates when a predefined fitness threshold ϵ is reached or when the maximum number of iterations T is completed. The integration of dynamic weight adaptation, a probabilistic two-phase search approach, and random reinitialization enables CDO to achieve reliable convergence to global optima. The overall flow is summarized in Figure 1 [42].

3.2 Chaotic Maps and the Logistic Map

Chaotic maps are simple yet powerful deterministic systems that exhibit complex and seemingly unpredictable behavior. These maps are widely used in metaheuristic algorithms to improve exploration and exploitation capabilities, particularly through randomization and sensitivity to initial conditions. One of the most commonly used

Algorithm 1 Cloud Drift Optimization (CDO)

Require: Population size N, maximum iterations T, search space bounds $[lb, ub]$, dimension d

Ensure: Best position $\mathbf{x}^*$, Best fitness $f(\mathbf{x}^*)$

Initialize population $X \in \mathbb{R}^{N \times D}$ within $[lb, ub]$ using Eq. (2)
Set initial random perturbation probability $z = 0.005$
for $t = 1$ to T **do**
    Evaluate fitness of all clouds
    Update best solution so far
    Update control parameters a, b, z based on iteration t
    for each cloud i **do**
        if rand $< z$ **then**
            Reinitialize $X(i, :)$ randomly using Eq. (7)
        else
            Compute $p = \tanh(|f(i) - \text{Best_fitness}|)$
            if rand $< p$ **then**
                Cloud Merging Mechanism (local refinement)
                Update position using Eq. (4)
            else
                Cloud Splitting Mechanism (global search)
                Update position using Eq. (5)
            end if
        end if
        Enforce bounds on $X(i, :)$
    end for
end forreturn Best position, Best fitness

chaotic maps in optimization is the Logistic Map, defined by the following recursive equation:

$$O_{t+1} = r \cdot x_t(1 - O_t), \quad 0 < O_t < 1, \quad 0 < r \leq 4, \tag{8}$$

where O_t is the state at iteration t, and r is the control parameter that dictates the dynamic behavior of the map. For values of r between 3.57 and 4, the Logistic Map exhibits chaotic behavior, making it particularly suitable for introducing diversity in population-based metaheuristics. The chaotic sequences generated can replace or enhance traditional random number generators to avoid premature convergence and enhance global search. In recent studies, chaotic maps like the Logistic Map have been employed in various metaheuristic algorithms for enhancing the exploration–exploitation trade-off. For instance, the Energy Valley Optimizer (EVO) by Azizi et al. [72] demonstrates the use of intelligent interaction dynamics in the population without direct chaos integration, while more recent works such as the Crane Search Algorithm with chaotic initialization [73] and the chaotic-enhanced Squirrel Search Algorithm [74] explicitly embed chaotic maps like the Logistic Map for better convergence behavior. Accordingly, the Logistic map will be integrated into our algorithmic framework to generate chaotic sequences, which aid in maintaining exploration capabilities throughout the optimization process. To visualize the effect of the

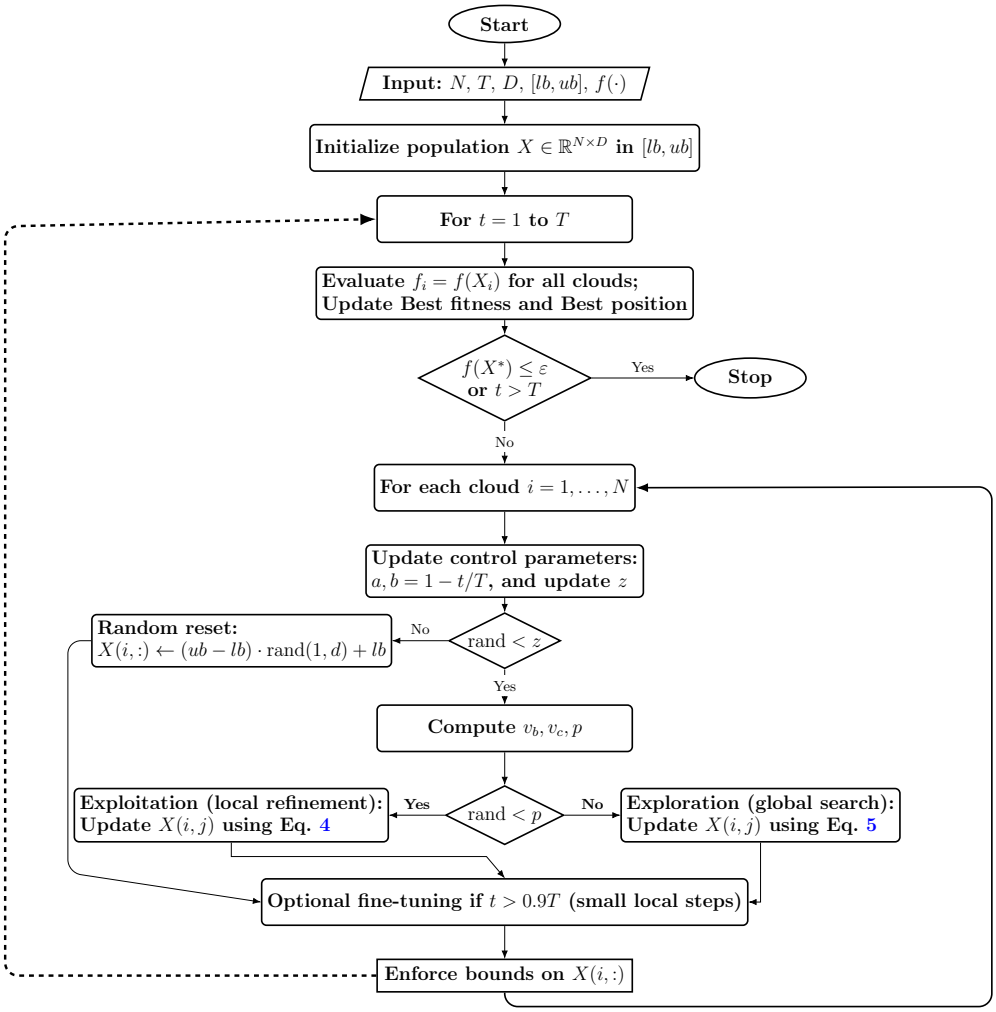


Fig. 1 Flowchart of the CAO algorithm

Logistic Map across different values of r, the map was simulated in MATLAB for $r = \{2.5, 3.2, 3.7, 4.0\}$ over 100 iterations. Figure 2 illustrates the behavior of the Logistic Map for various values of the control parameter r. For $r = 2.5$, the map converges to a fixed point. At $r = 3.2$, it enters a periodic cycle. As r approaches 4, chaotic behavior dominates.

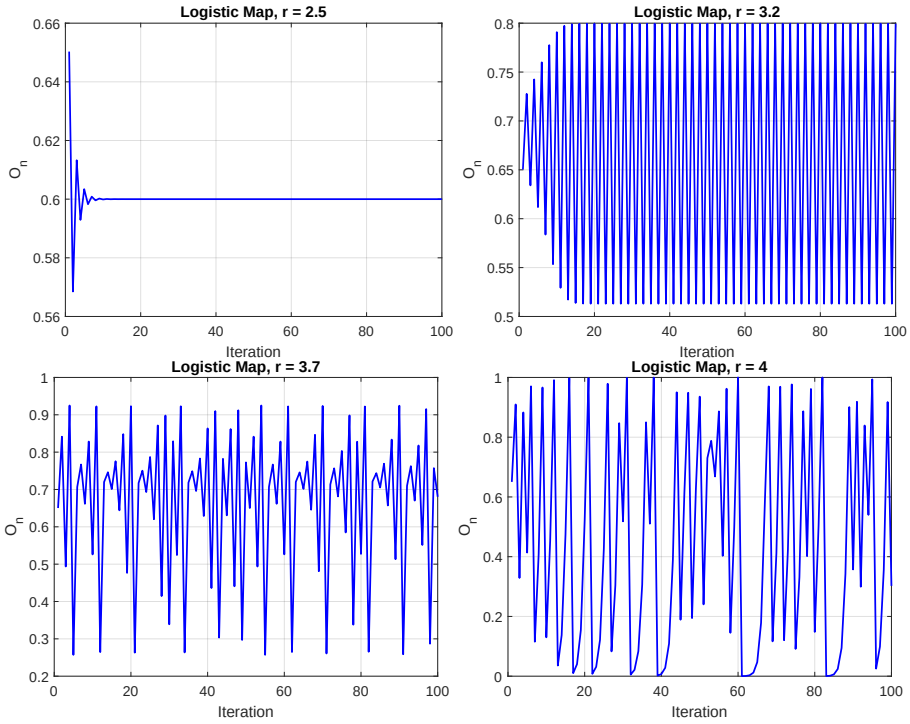


Fig. 2 Logistic Map outputs for different control parameter values: (a) $r = 2.5$, (b) $r = 3.2$, (c) $r = 3.7$, and (d) $r = 4.0$

4 The Proposed Chaotic Improved Cloud Drift Optimization

The proposed Chaotic Improved Cloud Drift Optimization (CI-CDO) is an enhanced version of the Cloud Drift Optimization (CDO) algorithm [42]. While CDO has shown competitive performance, its reliance on stochastic operators in the weight adaptation (Eq. 3) and position updating rules (Eq. 4–5) may lead to premature convergence in complex search landscapes. improves these aspects by introducing chaotic maps and an adaptive coefficient mechanism that strengthens exploration and exploitation.

4.1 Chaotic Parameter Control

In CDO, the transition parameters a and b (Eq. 4 and Eq. 5) decrease linearly over time. replaces these deterministic schedules with chaotic sequences $O(t)$ generated by logistic (Eq. 8):

$$O_{t+1} = r \cdot O_t \cdot (1 - O_t), \quad r = 4, \tag{9}$$

where $O_t \in (0, 1)$ and O_0 is a randomly chosen initial value in (0,1). This replacement ensures non-repetitive, ergodic variations of control parameters, increasing the likelihood of escaping local minima compared to the linear a and b used in CDO.

4.2 Cloud Dynamic Position Updating

Unlike the standard CDO, which updates candidate positions using two search operators (Eqs. 4 and 5), the proposed algorithm models cloud evolution using three atmospheric mechanisms: *Cloud Merging*, *Cloud Splitting*, and *Atmospheric Turbulence*. These mechanisms emulate the natural dynamics of cloud systems, where cloud formations continuously merge, split, and drift under varying atmospheric conditions. Collectively, they provide an adaptive balance between search diversification and solution refinement, resulting in a more effective and physically inspired position updating strategy.

4.2.1 Cloud Merging Mechanism

In the atmosphere, neighboring cloud fragments gradually merge under favorable environmental conditions, forming larger and denser cloud masses. This natural process concentrates cloud motion toward stable atmospheric regions and promotes the formation of coherent cloud structures. Inspired by this phenomenon, the proposed Cloud Merging mechanism guides promising cloud agents toward the best cloud formation while incorporating chaotic dynamics to improve convergence behavior. Accordingly, Eq. 4 is modified as follows:

$$X_{t+1}(i,j) = X_{\text{best}}(j) - cf \cdot |X_t(i,j) - X_{\text{best}}(j)| \cdot \cos(\theta), \tag{10}$$

where cf is a chaotic drift factor derived from the sequence O_t (Eq. 11).

Unlike the standard CDO, where deterministic coefficients regulate the convergence process, the proposed Cloud Merging mechanism employs chaotic dynamics to generate cf, introducing nonlinear and non-repetitive perturbations into the cloud drift process. These perturbations maintain the natural tendency of neighboring cloud masses to converge while reducing the risk of premature stagnation. Furthermore, the parameter θ represents an adaptive atmospheric approach angle selected through a roulette wheel mechanism. This adaptive strategy enables cloud agents to approach the best cloud formation from different directions, thereby improving convergence flexibility and increasing the probability of escaping narrow local optima [75].

The chaotic drift factor is computed as:

$$cf = O_t * b, \tag{11}$$

where the atmospheric scaling factor b is updated from the original CDO algorithm as:

$$b = S - \frac{S \cdot t}{T}, \qquad S = 2, \tag{12}$$

where t is the current iteration and T is the maximum number of iterations.

Fig. 3(Left) illustrates the evolution of the chaotic drift factor cf. Since b decreases linearly from $S = 2$ to 0, the amplitude of cf gradually diminishes throughout the optimization process. During the early iterations, the relatively large value of b allows broader cloud drift movements, enabling neighboring cloud fragments to merge while

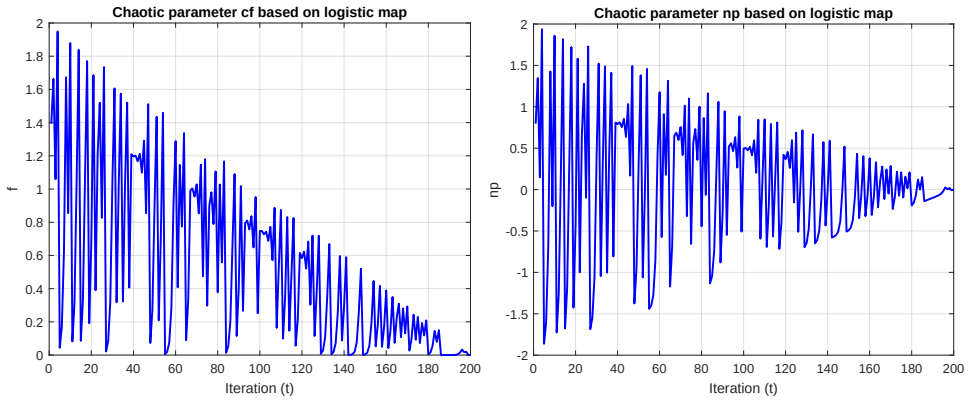


Fig. 3 Evolution of the atmospheric chaotic parameters generated by the logistic map: (Left) chaotic drift factor cf, and (Right) Atmospheric Turbulence Index np.

maintaining sufficient search diversity. As the optimization progresses, the reduction of b suppresses the chaotic oscillations of O_t, producing finer cloud movements that promote compact cloud formations around the best solution. Consequently, the proposed Cloud Merging mechanism provides accurate local refinement, stable convergence, and effective exploitation of promising search regions.

4.2.2 Cloud Splitting Mechanism

In the atmosphere, large cloud formations may split into smaller cloud fragments under the influence of atmospheric instability and wind shear. These separated cloud fragments drift independently, allowing clouds to spread over wider regions and increasing the probability of reaching new atmospheric zones. Inspired by this natural phenomenon, the proposed Cloud Splitting mechanism disperses cloud agents toward different search directions, thereby enhancing search diversity and reducing the likelihood of premature convergence. Accordingly, Eq. 5 is modified as follows:

$$X_{t+1}(i,j) = cf \cdot \big(X_t(k,j) - \gamma \cdot X_t(i,j)\big), \tag{13}$$

where $X(k,:)$ denotes a randomly selected cloud agent from the current population, $\gamma \in \{0,1\}$ is an atmospheric switching factor that adaptively regulates the influence of the current cloud position, and cf is the chaotic drift factor derived from the logistic chaotic map (Eq. 11).

Unlike the standard CDO, which relies on deterministic coefficients to diversify the search, the proposed Cloud Splitting mechanism incorporates chaotic dynamics into the cloud drift process. The chaotic drift factor enables cloud fragments to spread through irregular yet bounded movements, allowing them to explore unexplored regions while preserving the stability of the search process.

During the early iterations, the relatively large value of cf promotes wider cloud dispersion, increasing the probability of discovering promising regions across the search space. As the optimization progresses, the gradual reduction of the chaotic drift intensity naturally limits unnecessary dispersion and allows cloud fragments to concentrate

on more promising areas. Consequently, the proposed Cloud Splitting mechanism provides effective global exploration, maintains population diversity, and reduces the risk of premature convergence in complex multimodal optimization landscapes.

4.2.3 Atmospheric Turbulence Mechanism

Atmospheric turbulence represents the dynamic instability of cloud motion and serves as the transition controller between the Cloud Splitting and Cloud Merging mechanisms. Unlike the standard CDO, where the selection between the exploration and exploitation phases is determined by comparing a random number with the probability value p (Eq. 6), the proposed framework employs an adaptive *Atmospheric Turbulence Index* (np) to regulate the cloud evolution process. The operational mode is determined according to the condition $-1 \leq np \leq 1$, as illustrated in Figure 4.

The Atmospheric Turbulence Index is computed as:

$$np = (2 \cdot b)\, O_t - b, \tag{14}$$

where O_t is the chaotic sequence generated by the selected logistic chaotic map.

Fig. 3(Right) illustrates the evolution of the Atmospheric Turbulence Index. The parameter np oscillates within the interval $[-2, 2]$, providing an adaptive mechanism for regulating cloud evolution throughout the optimization process. When $np \in [-1, 1]$, atmospheric instability increases, activating the Cloud Splitting mechanism to disperse cloud agents over unexplored regions and enhance search diversity. Otherwise, the atmosphere becomes relatively stable, triggering the Cloud Merging mechanism to consolidate neighboring cloud agents around promising regions and improve local refinement. Consequently, the proposed turbulence-driven switching strategy establishes a smooth and adaptive transition between diversification and intensification, resulting in improved convergence stability and solution quality.

4.3 Multi-Strategy Update

One of the key differences between CDO and the proposed lies in the update mechanism. In the original CDO, solution refinement relied heavily on random perturbations and weight-based adjustments (Eq. 3), which could limit diversity and reduce the adaptability of the search process. To overcome these limitations, introduces a multi-strategy update that replaces the random perturbation phase with a structured *multi-candidate generation* mechanism guided by chaotic coefficients, see Algorithm 2.

4.3.1 Improved Coefficient Vector:

Instead of depending only on fitness-ranked weights, defines a dynamic coefficient vector $\mathbf{c}_i$ for each ith cloud that integrates nonlinear chaotic dynamics:

$$\mathbf{c}_i = \begin{bmatrix} c_{i1} \\ c_{i2} \\ c_{i3} \\ c_{i4} \end{bmatrix} = \begin{bmatrix} \bar{r}_1 \\ (1-b) + r_2 \\ \delta \cdot O_t \\ \bar{u} \cdot \bar{v}^2 \cdot \cos(2O_t \cdot u) \end{bmatrix} \tag{15}$$

where $r_1, r_2 \sim U(0,1)$, $\delta \in \{0,1\}$ is a binary vector switch and $u, v \sim \mathcal{N}(0,1)$. Compared with Eq. 3 in CDO, which updates weights primarily as a function of fitness ranking, the proposed formulation injects chaotic perturbations that enhance population diversity and mitigate premature convergence.

4.3.2 Multi-Candidate Generation:

To further strengthen the balance between exploration and exploitation, employs multiple candidate updates per individual, leveraging different subsets of the chaotic coefficient vector. Instead of a single perturbed update, each solution generates parallel candidate moves:

$$X'_t(i) = M_t(i) - \left(\beta_1 \cdot X_{\text{best}} - \lambda \cdot X_t^i\right) \cdot \mathbf{c}_k, \tag{16}$$

$$X''_t(i) = X_t(s_1) + \left(\beta_2 \cdot X_{\text{best}} - \beta_3 \cdot X_t(s_2)\right) \cdot \mathbf{c}_m, \tag{17}$$

where M is a mean-guided drift term for the ith cloud, $\beta_{1,2,3} \in \{1,2\}$ are integers random sampled, $X(s_1), X(s_2)$ are two random clouds selected from the current population, and $\mathbf{c}_k, \mathbf{c}_m$ are chaotic coefficient subsets randomly selected where $1 \leq k, m \leq 4$. This integer-based selection introduces discrete randomness into the update process, enabling more diverse exploration patterns compared with deterministic weighting. The perturbation vector λ is defined as:

$$\lambda = \bar{c} \cdot \exp(b)\,, \tag{18}$$

where $\bar{c} \sim \mathcal{N}(0,1)$ produces Gaussian noise vector and the exponential decay term gradually reduces the perturbation strength across iterations and the scaling factor b is calculated from Eq. 12. The drift vector M is defined as [75]:

$$\bar{M}_t = X_t(s_3) \cdot \lfloor \texttt{rand} \rfloor + \bar{X}_t(m) \cdot \lceil \texttt{rand} \rceil, \tag{19}$$

where $X(s_3)$ is a randomly selected solution from index range $[\lceil N/3 \rceil, N]$ and $\bar{X}(m)$ is the mean of a third of the population which randomly selected from the current population. To further diversify the search, applies a crossover-like operator exclusively to X'_i with probability 0.8. A subset of its dimensions is replaced with either random values within the bounds $[lb, ub]$ or values copied from a randomly selected peer solution with probability 0.2:

$$\text{If } rand > 0.8, \qquad X'_t(i,j) = \begin{cases} \text{rand}[lb_j, ub_j], & \text{with probability 0.2,} \\ X'_t(i,j), & \text{otherwise.} \end{cases} \tag{20}$$

This crossover-like step injects controlled randomness while maintaining structural inheritance, thus preventing premature stagnation and enhancing exploration capacity. The last step is to add those three generated candidates $\{X_t(i), X'_t(i), X''_t(i)\}$ as shown in Eqs. 10 or 13, 16 and 17, respectively, to the current population, which the best-performing N solutions is retained for survival. This *multi-strategy mechanism* ensures that does not rely on a single trajectory of random perturbations, but rather explores a richer neighborhood of solutions with chaotic diversity, improving robustness against stagnation while preserving convergence efficiency

Algorithm 2 Chaotic Improved Cloud Drift Optimization (CI-CDO)

Require: Population size N, maximum iterations T, bounds $[lb, ub]$, dimension D, objective function $f(\mathbf{x})$

Ensure: Best position X_{best}, Best fitness $f(X_{best})$

Initialize population $X \in \mathbb{R}^{N \times D}$ randomly within $[lb, ub]$ using Eq. (2)
Initialize chaotic sequence $O_{t...T}$ using Eq. (9)
Evaluate fitness $f_i = f(X_i)$ for all clouds and set the best cloud
for $t = 1$ to T **do**
    Update scaling factor b using Eq. (12)
    for each solution X_i **do**
        Generate coefficient vectors $\mathbf{c}_i$ using Eq. (15)
        Update the mean-guided drift term M using Eq. (19)
        Update scaling factor λ using Eq. (18)
        Compute probability of chaotic control factors f, np from Eqs. (11),(14)
        if $-1 \leq np \leq 1$ **then**
            % Cloud Merging Mechanism
            Update X_i using Eq. (10)
        else
            % Cloud Splitting Mechanism
            Update X_i using Eq. (13)
        end if
        Generate additional candidates X_i', X_i'' using Eqs. (16),(17)
        Apply boundary handling
        Calculate the fitness and add the candidate $\{X_i, X_i', X_i''\}$ to the current population
        Select top N fittest individuals from the new population $(N + 3)$
        Update best solution so far
    end for
end forreturn Best position, Best fitness

5 Experimental Framework and result analysis

To comprehensively evaluate the performance of the proposed Chaotic Cloud Drift Optimization (CI-CDO) algorithm, three distinct experimental scenarios were conducted. First, the algorithm was tested on the 50 function of CEC 2021 benchmark suite, which includes a variety of challenging unimodal, multimodal, hybrid, and composition functions designed to assess the global optimization capability of meta-heuristics. CI-CDO was compared against several state-of-the-art algorithms using standard metrics such as best, mean, worst and standard deviation across 30 independent runs. Secondly, the algorithm was applied to a set of 21 UCI benchmark datasets of varying sizes (small, medium and large) for binary feature selection, utilizing a wrapper-based approach with the KNN classifier as the evaluator. A transfer function was used to binarize continuous solutions, and the fitness function was defined as a weighted combination of classification error and the number of selected features. Across multiple datasets of varying dimensionality and complexity, CI-CDO achieved

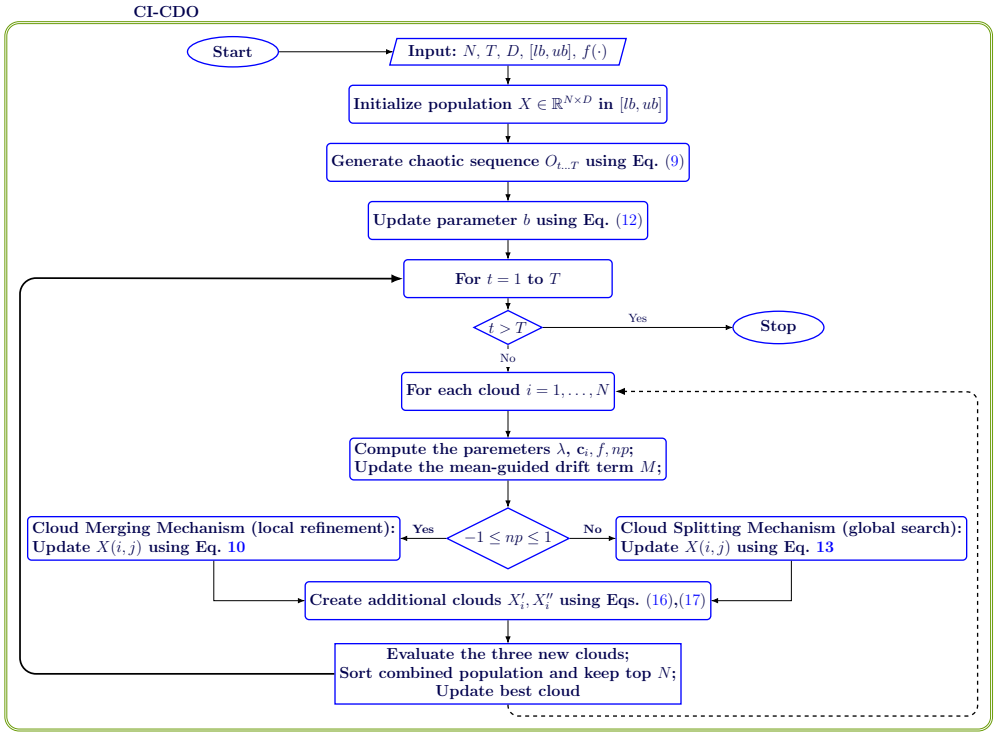


Fig. 4 Flowchart of the CICAO algorithm

high classification accuracy while selecting minimal feature subsets, outperforming other recent binary metaheuristic algorithms in both accuracy and reduction rate. Lastly, CI-CDO was applied to time-series forecasting on real-world seasonal water quality and temperature datasets collected from Egyptian monitoring stations. The forecasting task was formulated as a joint parameter tuning and feature optimization problem for an Artificial Neural Network (ANN) regression model, aimed at predicting seasonal variations in water temperature and quality indicators (e.g., pH, DO, TDS).

CI-CDO successfully optimized both the selection of relevant input variables and the ANN hyperparameters, resulting in superior predictive performance. In particular, the ANN– framework achieved lower RMSE and higher R^2 scores compared to baseline optimization methods. These findings highlight the robustness and adaptability of CI-CDO, demonstrating its effectiveness not only on standard mathematical benchmarks but also in real-world environmental forecasting applications. This study evaluates and benchmarks seven advanced metaheuristic optimization algorithms plus the basic algorithm CDO, Mountain Gazelle Optimizer (MGO) [75], Honey Badger Algorithm (HBA) [76], Fire Hawk Optimizer (FHO) [77], Flower Pollination Algorithm (FPA) [78], Dynamic Neighborhood Learning-based Particle Swarm Optimizer (DNLPSO) [79], Adaptive Gbest-guided Gravitational Search Algorithm (GGSA) [80], and an improved Grey Wolf Optimizer integrated with Cuckoo Search (GWO_CS) [81], across

three domains: synthetic function optimization (CEC 2021), feature selection on UCI datasets, and forecasting climate indicators in Egypt. The parameters of the shared algorithms are summarized in Table 1. All experiments except forecasting test were implemented using MATLAB R2019b on a PC with an Intel(R) Pentium(R) CPU G3220 @ 3.00GHz CPU and 8 GB RAM.

Table 1 Key Parameter Values Used in Each Algorithm

Algorithm	Parameter(s)	Value
MGO [75]	Parameter-free	–
HBA [76]	food-getting ability β, scaling in density factor C	$\beta = 6$, $C = 2$
FHO [77]	Number of Fire Hawks (HN), Random factor α	$HN \sim \text{randint}[1, \lceil nPop/5 \rceil]$, $\alpha \sim \text{randint}(1, nPop2)$
FPA [78]	Switch probability p	$p = 0.3$
DNLPSO [79]	Inertia weight, Acceleration coeffs, Learning freq., Shuffle interval	$w_0 = 0.9$, $w_1 = 0.4$; $c_1 = c_2 = 1.49445 \cdot \text{rand}()$; $m = 3$; $g = 5$.
GGSA [80]	Gravitational constant, Distance norms, Velocity coeffs	$G_0 = 100$, $\alpha = 20$; $R_{\text{norm}} = 2$, $R_{\text{power}} = 1$; $C_1 = 1 - (t^3/T^3)$, $C_2 = t^3/T^3$
GWO_CS [82, 83]	Lévy exponent, Step scaling	$\beta = 1.5$; $s = 0.001$

6 Experimental of CEC 2021 Benchmark Functions

To evaluate the performance of the proposed algorithm, we adopted the benchmark functions provided by the CEC 2021 Special Session and Competition on Single Objective Bound Constrained Numerical Optimization [84].

This benchmark suite comprises 50 diverse test functions, categorized into unimodal, multimodal, hybrid, and composite types. These functions are specifically designed to challenge various aspects of optimization algorithms, including convergence speed, robustness, and ability to escape local optima.

Each function is defined within a bounded search space, typically $[-100, 100]^D$, and includes a well-defined global minimum. The benchmark includes rotation and shifting mechanisms to prevent algorithms from exploiting symmetry or origin-centered search spaces. Standard evaluation protocols are used, including a fixed number of function evaluations (1×10^6) and multiple independent runs to ensure statistical reliability. Each algorithm was executed with a population size $N = 100$ over 30 independent runs.

The full description and implementation of these functions are publicly available in the technical report and GitHub repository maintained by the competition organizers [84].

To assess the performance of the compared algorithms, several metrics were used. These included the best, worst, and mean fitness values obtained over 30 independent runs, along with the standard deviation (STD) to measure the robustness of each algorithm. The best results for each function are highlighted in bold. To evaluate the statistical significance of performance differences, a Wilcoxon signed-rank and Friedman tests were applied at a 0.05 significance level.

6.1 Results and discussion

To further validate the robustness and convergence capabilities of the proposed metaheuristic algorithms, each method was benchmarked against the 50 test functions from the CEC 2021 constrained optimization suite. The following Tables 2–7 summarize the results obtained on the CEC 2021 by the compared algorithms over 30 independent runs.

All tested metaheuristic algorithms CI-CDO, CDO, MGO, HBO, FHO, FPA, DNLPSO, GGSA, and GWO_CS were evaluated under identical conditions. Table 2 presents the minimum objective values achieved, with results reported for all algorithms. Boldface indicates the best-performing algorithm(s) for each function in terms of achieving the lowest (i.e., best) objective value. These results illustrate that CI-CDO consistently outperformed other methods across a wide range of problem categories, especially in shifted and rotated landscapes. Notably, most algorithms showed similar convergence on some simpler problems (e.g., F1–F4), while others like F15, F19, F65 and F47 revealed significant performance gaps due to constraint handling or landscape complexity.

To assess the optimization performance of the proposed and comparative algorithms, each was independently executed 30 times, and the average of the best objective values is reported in Table 3.

Among all tested algorithms, the propose CI-CDO algorithm consistently achieved superior performance across the majority of functions, often obtaining near optimal objective values. It outperformed other methods particularly in unimodal and multimodal landscapes, demonstrating strong convergence capabilities and stability.

Notably, algorithms such as MGO, HBA, and FHO showed promising performance on simpler functions but were less consistent across complex, constrained landscapes. Classical metaheuristics like DNLPSO and GGSA exhibited higher average objective values, indicating challenges in navigating the constraint boundaries efficiently. The hybrid GWO_CS performed moderately well in certain instances but lacked robustness on functions with highly nonlinear constraints.

To complement the average performance evaluation, we also report the standard deviation of the best-found objective values across 30 independent runs in Table 4. The standard deviation reflects the stability and robustness of each algorithm under stochastic variations.

A lower standard deviation indicates consistent performance, while higher values suggest that the algorithm's results vary significantly across different runs, potentially due to premature convergence or difficulty in handling constraints. As shown in Table 4, the proposed CI-CDO demonstrates superior robustness, achieving near-zero standard deviation on many functions, especially the simpler and unimodal ones (e.g., F1–F5, F8, F9). It maintains reasonable stability even on more complex functions, outperforming most of the comparative algorithms.

Algorithms such as DNLPSO and FHO exhibit high variance on several benchmarks, implying unstable behavior or sensitivity to initial conditions. The GWO_CS algorithm shows moderate consistency but fails to match the robustness of CI-CDO across the full suite. To further analyze algorithm robustness, Table 5 presents the

Table 2 Minimum objective values achieved by each algorithm on the CEC 2021 benchmark functions.

Fun	CI-CDO	CDO	MGO	HBA	FHO	FPA	DNLPSO	GGSA	GWO_CS
F1	**0**	**0**	**0**	**0**	1.3E-268	8.25E-21	1.52E-16	2.88E-17	**0**
F2	**0**	**0**	**0**	**0**	58.91605	158.8961	30.02067	0.031228	**0**
F3	**0**	**0**	**0**	**0**	3.38E-09	19.71842	19.89917	1.17E-25	**0**
F4	**0**	**0**	**0**	**0**	**0**	1.673865	1.214083	2.515431	**0**
F5	**0**	**0**	1.01E-95	**0**	284.0637	5.971598	193.0337	1.81E-17	**0**
F6	**-2.22E-16**	0	3.4E-08	-2.2E-16	22.72276	3.404414	7.14461	0.177917	0.000605
F7	**-2.22E-16**	-1.11E-16	3.97E-08	6.87E-12	84.22466	1.325863	57.04746	0.02177	0.000449
F8	**0**	**0**	**0**	**0**	**0**	150.8964	0.667535	**0**	**0**
F9	**0**	**0**	**0**	**0**	9.9E-202	1.09E-11	1.02E-09	4.73E-12	8.88E-15
F10	**0**	**0**	0.001679	2.94E-06	0.616416	49.14935	48.89981	0.001657	0.000407
F11	**0**	1.122E-07	1.22E-22	**0**	1.77E+09	1.01E-20	2772.915	0.006286	2677.35
F12	**0.2185992**	132.06957	132.2937	248.1405	3739.432	297.2006	401.1281	1666.726	484.0882
F13	24.162778	20.16173	36.21995	25.72519	182.6858	34.46556	**7.979981**	20.16173	41.88833
F14	1.0405967	**0.1399194**	1.446911	1.008868	29.73568	1.915401	1.409958	2.24709	1.832726
F15	**0.1040852**	2.1980892	33.66597	378.074	325521	17.96974	242.2296	162.1461	2233.062
F16	**0.2458135**	0.7291385	0.847503	7.759555	698.3991	6.099351	34.56481	713.2951	245.7819
F17	**0.2562383**	0.7397218	0.377041	30.7672	32486.01	0.837691	15.69661	361.5004	3563.16
F18	100	100.00027	100	100	398.4857	**0.021188**	100.0835	100	107.2216
F19	402.18021	300.00205	404.1427	411.755	544.7375	**100.0237**	458.7315	200	300.0484
F20	400	400.00168	485.6894	400	704.1836	**205.8855**	485.6891	400	456.9573
F21	**0**	1.5381741	0.946812	0.0014	2E+09	0.027029	15948.37	0.010056	235.7331
F22	**25.211393**	131.31604	377.0142	238.2373	3708.693	652.0471	369.2441	1967	296.8539
F23	31.097319	24.413946	30.45305	30.67596	170.9566	45.2332	33.3823	**20.97469**	45.13869
F24	**0.7386925**	1.2952723	1.258614	1.272451	28.41121	2.0451	2.761399	2.177048	1.23422
F25	402.63493	2770.1845	8758.807	991.4533	86270.06	**209.6626**	10327.2	81970.58	22451.68
F26	**1.5818863**	1.9072324	7.875614	7.910846	658.1441	6.327422	11.38486	634.2685	139.6819
F27	**13.947858**	393.16848	2313.534	787.537	72135.84	98.07669	3511.957	26784.72	2801.931
F28	**100**	100.00026	**100**	**100**	483.0445	100	100.0037	100	102.9049
F29	436.47724	464.00943	415.0635	433.6634	516.9655	**100.0328**	437.0642	476.4599	364.0766
F30	402.75075	410.60383	413.7929	410.2024	535.3276	**399.0223**	410.4177	413.6719	419.5184
F31	**100**	100	**100**	**100**	1.7E+09	**100**	2140.395	100.0052	2437.873
F32	**1103.4463**	1221.9783	1374.933	1237.137	4109.894	1324.965	1538.278	2707.485	1353.573
F33	724.47886	720.16173	731.5867	730.0423	867.0638	736.5687	**717.8518**	721.39	738.8596
F34	1901.0775	**1900.5627**	1901.165	1901.094	1934.477	1902.111	1902.258	1902.563	1901.704
F35	1701.3072	**1700.2082**	1708.78	1958.406	216652.7	1717.951	1993.017	2255.316	3855.581
F36	**1600.3384**	1600.6855	1606.183	1601.614	2196.777	1603.958	1637.217	2175.209	1741.594
F37	2100.5054	**2100.2237**	2100.296	2130.458	91794.71	2100.832	2139.48	2460.792	5122.71
F38	**2300**	2300.0003	**2300**	**2300**	2592.859	2300	2300.458	**2300**	2305.702
F39	2803.6084	2700.0023	2805.266	2809.274	2920.687	**2500.006**	2840.8	2600	2776.25
F40	2900	2900.0017	2900	2900	3196.884	**2701.026**	2985.717	2900	2992.67
F41	**100**	112.21831	100.356	100.0009	2.28E+09	100.0346	52423.59	100.0001	361.0121
F42	**1119.3901**	1359.7207	1466.982	1377.505	4895.508	1628.442	1478.805	2948.35	1579.269
F43	741.74281	722.86351	743.6205	737.2507	868.6762	745.6048	722.0513	**721.5897**	740.2909
F44	1901.0773	**1900.9167**	1901.271	1901.329	1940.591	1901.757	1902.808	1901.866	1901.773
F45	**2006.9004**	3051.0421	3967.795	3176.776	151171.9	2061.304	6030.213	65815.4	14334.63
F46	**1601.5878**	1602.3964	1607.724	1603.141	2102.059	1602.959	1629.511	1865.352	1678.164
F47	**2139.6605**	2855.9599	3568.087	2769.516	55839.18	2185.622	4194.766	37877.28	4572.127
F48	**2300**	2300.0002	**2300**	**2300**	2612.102	2300	2300.169	**2300**	2303.975
F49	2824.8119	2882.8061	2817.08	2821.876	2916.142	**2500.007**	2843.401	2867.927	2806.126
F50	2902.7989	2910.5541	2900.141	2902.096	3081.605	**2899.042**	2910.516	2913.657	2924.444

worst-case objective function values obtained by each algorithm across 30 independent runs on the CEC 2021 test suite. The worst-case result highlights the maximum degradation in solution quality an algorithm may suffer. As seen in Table 5, the proposed CI-CDO consistently achieves the best worst-case performance (bolded) across most functions. It either matches or outperforms all comparative algorithms, especially in simpler functions (F1–F10) and even in many complex scenarios (e.g., F15, F22,

Table 3 Average best objective values achieved by each algorithm on the CEC 2021 benchmark functions.

Fun	CI-CDO	CDO	MGO	HBA	FHO	FPA	DNLPSO	GGSA	GWO_CS
F1	**0**	**0**	**0**	**0**	1.4E-248	2.47E-20	6.74E+08	0.002279	**0**
F2	**0**	**0**	**0**	**0**	646.037	308.8016	1109.516	43.52266	105.3724
F3	**0**	**0**	**0**	**0**	15.99416	31.83783	135.2118	2.155745	32.50135
F4	**0**	**0**	**0**	**0**	**0**	2.372857	7781.723	3.567253	0.361255
F5	**0**	**0**	1.24E-48	**0**	495.3437	16.07504	146252.5	74.1717	0.013437
F6	8.017E-12	**0**	0.000664	1.3E-05	155.2287	8.674883	210.807	0.8195	0.023624
F7	4.783E-09	**-2.22E-17**	0.012859	5.48E-06	175.088	2.098051	29322.65	51.20701	0.01984
F8	**0**	**0**	**0**	**0**	**0**	410.4666	430.5592	17.78485	**0**
F9	**0**	**0**	**0**	**0**	1.15E-39	4E-11	42.38153	7.58E-12	8.88E-15
F10	**0**	**0**	0.00464	0.000308	14.28352	49.41153	93.32044	33.23464	49.29849
F11	3.458E-22	4.853E-07	1.29E-21	**6.1E-23**	2.52E+09	3.2E-20	2.52E+09	0.015511	81569265
F12	**49.550446**	466.58351	601.6609	928.3648	4198.456	459.4633	1116.491	2399.093	899.0995
F13	34.20726	**20.161731**	46.59279	44.2343	204.4986	41.38475	147.3137	23.02688	65.58879
F14	2.1815997	**1.0396249**	2.988996	1.803194	42.79109	2.737963	1304.651	3.369731	4.8759
F15	**10.309281**	110.73093	249.5156	1585.874	815076.7	32.6454	528277.4	844.9024	95680.36
F16	**1.2907661**	216.02196	126.4596	312.2477	911.7811	15.89559	291.3732	958.0004	428.2817
F17	3.4945392	93.062662	103.7784	700.9949	227635.6	**2.09972**	6085.932	743.3212	43564.73
F18	100.74184	132.99158	100.9362	100.0921	495.9366	**94.16723**	484.6055	100	125.7812
F19	408.80646	338.05496	415.2538	438.7952	580.8583	**253.6095**	509.1551	266.6667	431.2723
F20	465.5082	428.628	485.9038	445.797	759.4244	**393.5295**	716.0012	471.4204	517.1041
F21	**7.792E-21**	3967.8878	1777.592	61.19109	2.93E+09	0.091215	3.15E+09	140.9847	1.66E+08
F22	**489.39562**	595.47071	1035.734	1303.92	4343.644	858.1731	1318.841	2557.46	1128.23
F23	46.332187	37.131171	60.15792	58.63596	205.0649	56.28955	117.1133	**23.4072**	62.19483
F24	2.556004	3.1305753	3.057229	**2.37621**	50.10316	2.895173	25575.94	3.033408	5.98284
F25	1312.0845	71076.025	42481.04	5025.131	246010.5	**496.575**	173812.8	114526.5	233775.9
F26	**4.7428839**	204.67476	136.5291	288.8861	964.8452	17.53992	292.771	930.7863	364.9121
F27	301.12234	18783.425	13933.1	1752.001	412879.4	**246.9553**	79831.07	57865.29	45261.58
F28	100.69204	834.43283	100.756	202.1526	600.1384	100	736.5823	**100**	152.1429
F29	487.49437	579.57895	443.4412	492.8823	546.5569	**319.3513**	491.6901	524.6687	428.9999
F30	441.87359	449.97043	456.5218	440.8888	615.0714	**402.4179**	750.4473	415.2025	472.8638
F31	**100**	100	**100**	100	2.61E+09	**100**	3.96E+09	100.0141	48443536
F32	**1128.5429**	1511.7499	1696.655	2042.704	5259.876	1586.342	2483.872	3379.527	2107.023
F33	734.03896	**720.16173**	744.5038	745.2964	903.6733	740.5702	785.8667	723.128	761.3892
F34	1902.1972	**1900.9632**	1902.654	1901.924	1949.177	1902.707	10056.36	1903.6	1905.87
F35	**1708.764**	1801.0431	1891.622	3186.071	861571.4	1731.94	223459.3	2551.068	77865.64
F36	**1601.5614**	1787.2157	1719.936	1943.746	2544.33	1615.271	1892.095	2555.775	1997.592
F37	2105.4456	2202.0842	2206.592	2863.988	243017.1	**2102.148**	232910.1	2773.262	38453.1
F38	2300.6807	2350.3288	2300.792	2300.177	2710.055	2300	2864.816	**2300**	2319.597
F39	2808.3944	2744.9908	2816.689	2838.247	2968.552	**2667.837**	2910.226	2670	2821.601
F40	2970.578	2922.9209	2983.002	2942.907	3257.329	**2893.368**	3112.31	2979.99	3013
F41	**100**	5023.4353	2311.033	128.7733	2.97E+09	100.1058	4.43E+09	233.6913	94949794
F42	**1547.5292**	1794.8397	2145.788	2408.481	5599.801	1924.183	2440.248	3557.083	2164.07
F43	753.9929	734.60395	762.1511	755.2655	906.5788	754.9259	789.2048	**723.6576**	766.7208
F44	1902.485	1902.7304	1903.111	**1902.404**	1956.011	1902.912	40732.05	1903.068	1906.453
F45	2857.2085	68069.649	41046.3	7304.393	284593.3	**2258.208**	99879.96	107215.7	170332
F46	**1603.907**	1835.885	1716.928	1894.671	2544.944	1619.222	1921.791	2507.798	2012.766
F47	2412.8462	29257.396	13973.37	4015.427	436036.5	**2331.191**	43599.48	64776.6	54065.42
F48	2300.706	3302.041	2300.567	2406.021	2830.276	2300	3023.565	**2300**	2333.95
F49	2883.7512	2987.2072	2839.327	2886.657	2954.045	**2704.825**	2899.807	2924.418	2836.188
F50	2934.8322	2949.1054	2968.421	2942.092	3115.579	**2903.121**	3144.829	2913.674	2969.677

F26, F32, F46). On the other hand, algorithms like DNLPSO, FHO, and GWO_CS suffer from significantly higher worst-case values, indicating performance collapse in some runs potentially due to premature convergence, poor exploration, or sensitivity to problem landscape.

Table 4 Standard deviation of best objective values over 30 runs for each algorithm on the CEC 2021 benchmark functions.

Fun	CI-CDO	CDO	MGO	HBA	FHO	FPA	DNLPSO	GGSA	GWO_CS
F1	**0**	**0**	**0**	**0**	4.1E-247	5.69E-20	1E+10	0.01014	**0**
F2	**0**	**0**	**0**	**0**	1515.21	523.209	3589.996	217.2958	3160.941
F3	**0**	**0**	**0**	**0**	76.75069	46.21126	377.2482	5.969754	124.2863
F4	**0**	**0**	**0**	**0**	**0**	3.183469	155967.4	4.30273	4.501819
F5	**0**	**0**	3.22E-47	**0**	671.2339	26.96923	3739801	436.7193	0.40312
F6	1.094E-10	**0**	0.003067	0.000276	400.7597	15.1558	843.4256	1.426196	0.104925
F7	1.435E-07	**0**	0.07648	6.54E-05	232.0166	5.058986	173931	434.6574	0.058573
F8	**0**	**0**	**0**	**0**	**0**	834.8186	3861.956	103.1214	**0**
F9	**0**	**0**	**0**	**0**	1.21E-38	1.53E-10	330.0114	1.07E-11	8.88E-15
F10	**0**	**0**	0.007083	0.000779	89.74888	49.65986	332.1725	86.11313	82.56815
F11	1.578E-21	2.701E-06	4.52E-21	**4.54E-22**	3.13E+09	6.89E-20	1.93E+10	0.030059	5.8E+08
F12	**251.4307**	835.09581	1077.29	2307.48	4703.272	621.4543	2899.688	3184.891	1288.746
F13	45.820458	**20.161735**	64.59737	62.47481	236.23	49.39931	541.7479	26.63743	103.8841
F14	3.8774478	**1.8400129**	5.554242	3.436741	76.93613	3.723068	24449.52	4.293524	16.26655
F15	**33.770056**	262.07463	551.0609	14591.75	1422309	71.84998	6916514	1681.495	1257363
F16	**6.3890456**	477.29699	493.1248	854.4318	1282.658	33.68088	1159.492	1228.04	795.8633
F17	9.9420834	149.41904	426.4905	2017.727	430892.2	**4.759939**	78264.62	1223.276	667037.2
F18	102.43337	593.97467	102.2875	101.4886	584.8755	100	2746.944	**100**	209.9714
F19	419.2246	399.65798	425.7394	481.2606	694.4192	412.5272	696.4222	**300**	522.3834
F20	476.37239	486.94174	486.1496	486.3223	797.3869	**400**	2557.663	485.7278	560.2914
F21	**1.157E-19**	12059.211	8791.643	608.9228	3.54E+09	0.189793	3.28E+10	1408.588	1.64E+09
F22	**1038.3876**	1299.6003	1662.39	2852.364	4972.732	1071.06	4095.476	3111.206	1929.293
F23	65.835706	50.917142	105.4267	94.71483	226.9685	73.50283	557.4122	**29.88956**	108.4237
F24	4.8158509	6.4760312	5.250615	**3.751837**	68.82825	3.790993	599127.2	4.298723	27.08369
F25	6548.746	164705.75	125343.3	15434.54	412613.6	**742.0023**	1119784	173251.8	1389415
F26	**13.504597**	479.28305	386.6868	738.2686	1288.897	42.34437	1515.005	1247.585	718.7444
F27	747.81723	75249.112	28377.19	5207.693	880680.1	**482.8441**	1400729	100118.3	229247.3
F28	101.74298	3319.8978	102.6946	3146.7	690.035	100	2557.423	**100**	334.9693
F29	570.22526	683.33422	488.5683	634.7303	652.72	**483.8208**	597.2564	577.4336	499.8863
F30	495.18894	515.43096	507.0912	501.9528	654.1378	**410.3425**	2288.108	459.3773	521.833
F31	**100**	100	**100**	100.0001	3.47E+09	**100**	1.53E+10	100.024	5.18E+08
F32	**1347.9143**	1710.2621	2318.708	3250.306	5758.106	1759.513	3823.268	4127.134	3001.928
F33	746.08252	**720.16173**	757.4895	783.1121	928.0812	747.2547	993.301	725.6892	793.1875
F34	1905.3348	**1901.4003**	1905.499	1903.076	1963.14	1903.638	98413.06	1904.71	1946.385
F35	**1732.5669**	1963.0696	2128.129	11112.75	1724588	1758.518	1782782	3034.1	499727
F36	**1611.7012**	1959.9941	1856.662	2766.697	2866.815	1639.037	2663.604	3105.654	2339.422
F37	2129.5289	2277.9738	2546.845	5653.198	544893	**2103.607**	5982174	3076.349	358562.3
F38	2302.7118	2827.0385	2302.536	2301.466	2781.939	2300	5228.744	**2300**	2370.401
F39	2814.2904	2801.546	2846.665	2901.143	3050.62	2813.44	2985.155	**2700**	2902.446
F40	2976.5732	2986.9039	2985.973	2985.954	3300.444	**2900**	3954.835	2985.712	3059.948
F41	**100**	12056.112	10108.67	299.0833	3.53E+09	100.2239	1.93E+10	794.7662	1.6E+09
F42	**2068.34**	2258.1096	2806.749	4105.294	5997.783	2170.985	5164.27	4406.377	2954.771
F43	775.93587	748.6161	782.1017	790.1645	925.1631	769.3949	1100.567	**725.5606**	842.3687
F44	**1903.7788**	1906.5203	1906.807	1903.975	1975.509	1904.048	638474.9	1904.46	1943.424
F45	7167.5982	174092.78	110340.4	15715.56	431890.2	**2515.158**	405544.2	146250.5	441667.5
F46	**1611.9381**	1959.6949	1873.97	2836.894	2962.109	1651.45	2570.974	2931.76	2342.516
F47	2669.5592	101847.38	50680.53	9829.706	1645689	**2516.492**	238357.3	100396.6	168459.7
F48	2302.7314	6070.6911	2301.578	5466.381	2925.609	2300	5331.151	**2300**	2394.797
F49	2961.7037	3081.7814	**2867.927**	3117.865	3056.316	2912.183	2996.895	2991.928	2884.259
F50	3001.6641	2996.9146	3009.646	3004.3	3147.471	2913.749	5181.995	**2913.69**	3008.368

6.2 Statistical Analysis and Convergence Behavior

To comprehensively evaluate the performance of the proposed CI-CDO, we analyzed its optimization quality against seven state-of-the-art algorithms using a non-parametric statistical ranking (Friedman test), Wilcoxon signed-rank test, boxplots of solution distributions, and convergence behavior across 50 CEC 2021 benchmark functions (D=20).

6.2.1 Friedman test results

The Friedman test results, presented in Table 6, provide average rank-based comparisons of all algorithms. As shown in the table, CI-CDO achieves the lowest average rank of 2.56, which is significantly better than all other competing algorithms. This indicates that CI-CDO is the most robust and consistent algorithm across the entire suite of benchmark problems. The original CDO achieves an average rank of 3.81, which demonstrates that the chaotic enhancement in CI-CDO leads to a substantial performance gain. Other algorithms such as MGO (4.28), HBA (4) and FPA (3.75) exhibit moderate performance, while DNLPSO (7), GWO_CS (5.96), and FHO (8.19) show relatively weaker performance across the majority of functions. Notably, CI-CDO achieves the best or tied-best rank on many functions, including unimodal, multimodal, and hybrid categories, indicating strong convergence behavior and global search capability. These results are statistically meaningful and demonstrate the effectiveness of the proposed chaotic strategy in improving the exploration-exploitation balance of the base CDO algorithm.

6.2.2 Wilcoxon signed-rank test results

The Wilcoxon signed-rank test is a non-parametric statistical method widely used to compare two related samples or repeated measurements on a single sample to assess whether their population mean ranks differ [85, 86]. Using the Wilcoxon signed-rank test on the experimental data provided, where each feature's performance between pairs of algorithms was compared, the outcomes were summarized symbolically with corresponding rank statistics (R) as ✓ denoting a *win* (statistically significant better performance), ≅ indicating *no significant difference* (equivalence), and ✗ indicating a *loss* (statistically worse performance). The Wilcoxon test results affirm that while many features exhibit statistically significant improvements in certain algorithms, the existence of ties and losses underlines the importance of context-specific evaluation. These findings align with the best practices in statistical hypothesis testing for algorithm comparison, where non-parametric tests like Wilcoxon offer robust insights beyond simple mean comparisons [87]. Table 7 presents the results of the Wilcoxon signed-rank test comparing the proposed algorithm against seven competing methods (CDO, MGO, HBA, FHO, FPA, DNLPSO, GGSA, and GWO_CS) across the 50 benchmark functions of the CEC 2021 test suite. From the results, it is evident that most algorithm comparisons show a predominance of wins ✓, suggesting that for many features, the tested algorithms significantly outperform their counterparts. For instance, across methods such as FHO, FPA, DNLPSO, and others, p-values are consistently below the typical significance threshold (0.05), accompanied by positive rank statistics ($R = 1$), reinforcing strong evidence in favor of the first method being superior on those features. Conversely, there are occasional equalities (≅) and losses ✗, highlighting cases where differences are either statistically negligible or the tested algorithm underperforms. For example, in some features such as F11 and F29, losses ✗ occur with corresponding negative rank statistics ($R = -1$), signaling that the alternate algorithms outperform the baseline on those specific metrics. The balance of these outcomes provides a nuanced understanding of the comparative strengths and

weaknesses among the tested methods, guiding further algorithmic refinement. The last three rows of the table summarize the number of functions where the proposed method significantly outperformed, tied with, or was outperformed by each algorithm. Notably, the proposed method achieved significant superiority over FHO in 48 functions, GGSA in 47, and GWO_CS in 44. It also recorded 34 wins over MGO and 29 over FPA, indicating its consistent performance. These outcomes highlight the robustness and competitiveness of the proposed algorithm across a diverse set of optimization problems.

6.2.3 Boxplot and Convergence Curves

The boxplots and convergence figures for each benchmark function visually reinforce the Friedman ranking results. The proposed CI-CDO algorithm demonstrates, through the boxplot and convergence figures 5,6, its ability to maintain a good balance between exploration and exploitation. In comparison, algorithms like FHO and DNLPSO exhibited wider boxplots with larger interquartile ranges and higher variance, signifying instability and less reliable performance. The experimental results reveal varying degrees of convergence behavior, constraint-handling efficiency, and robustness under different transformation scenarios. In particular, GWO_CS and DNLPSO consistently demonstrated strong performance on rotated and translated problem instances, while HBO was more effective on simpler, unimodal landscapes. These results also highlight the effectiveness of the introduced multi-strategy technique in enhancing the performance of the original CDO algorithm. Overall, based on the test results, boxplot distributions, and convergence profiles, the CI-CDO algorithm exhibits superior performance, faster convergence, and greater solution stability compared to both conventional and recently developed metaheuristic algorithms. These outcomes confirm the effectiveness of the chaotic enhancement strategy and its ability to generalize well across a diverse set of optimization problems.

7 Experimental of Feature Selection Using KNN Classifier

In this study, feature selection is performed using a wrapper-based approach as a second experiment, where the classification performance is evaluated using the KNN algorithm. For each dataset, features and corresponding class labels are loaded, and the dimensionality of the dataset is determined. The experiments are repeated across 20 independent runs with 100 maximum iteration to ensure robustness and reliability of the results. In the second test, the algorithms were applied to wrapper-based feature selection using KNN classifiers over benchmark UCI datasets [88]. Each algorithm searches for the optimal subset of features that minimizes classification error. Each individual in the metaheuristic population is represented as a binary mask over the input features, where a value of 1 denotes feature inclusion. The wrapper evaluates each candidate using KNN classifier, and classification accuracy and selected features on the testing set is used as the fitness measure.

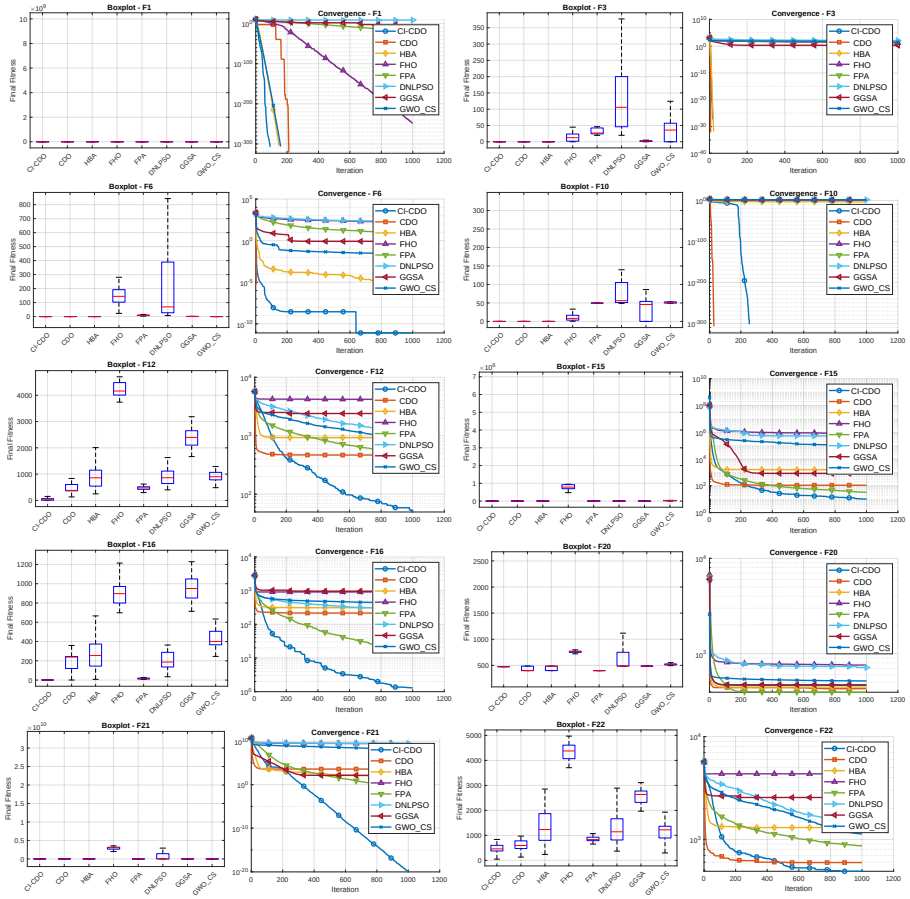


Fig. 5 Convergence curves of the compared algorithms for CEC21

Classification and Fitness Evaluation

To evaluate the classification performance, the dataset is partitioned into training and testing subsets. This splitting is performed using a hold-out method, where 70% of the data is used for training and the remaining 30% is used for testing. Importantly, the indices used for training and testing splits are saved during this process and consistently reused in all experiments. This ensures that all feature selection algorithms are evaluated on the exact same data partitions, enabling fair and unbiased comparisons. Once the training and testing sets are prepared, a KNN classifier is trained using the training set, with the number of neighbors set to $k = 5$ and the Euclidean distance metric as the similarity measure. The trained model is then applied to the test set to predict class labels. The performance is quantified using the misclassification rate, computed as the proportion of incorrectly classified instances in the test set. To evaluate the effectiveness of selected feature subsets, a fitness function is employed that balances classification accuracy with the number of selected features. The function is defined as a weighted combination of the misclassification rate and the ratio of selected

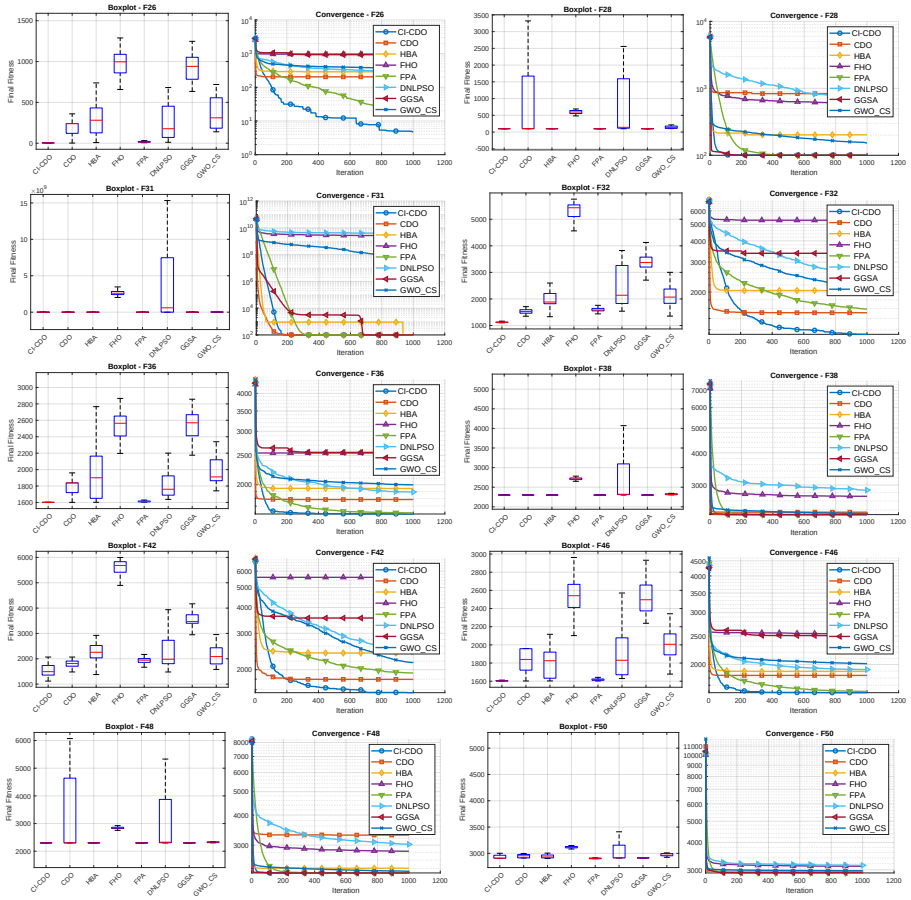


Fig. 6 Convergence curves of the compared algorithms for CEC21

features to the total number of features. Specifically:

$$\text{Fitness} = \omega \cdot \text{Error Rate} + (1-\omega) \cdot \left(\frac{\text{Number of Selected Features}}{\text{Total Features}} \right), \qquad \omega = 0.99.$$

This formulation prioritizes classification accuracy while still penalizing large feature subsets to encourage dimensionality reduction. This method provides a reliable framework for evaluating the trade-off between feature reduction and classification performance across different datasets and feature selection algorithms.

7.1 Transfer Function for Binary Conversion

In the proposed CI-CDO algorithm, a *transfer function* is employed to transform continuous solution vectors into binary representations. This step is essential for adapting the algorithm to discrete feature selection problems, where each feature is either selected (1) or discarded (0). Let $X(i,:)$ represent the continuous position vector of the

i-th agent. A modified hyperbolic tangent function is applied element-wise to produce a probabilistic selection likelihood [32, 89]:

$$S_x(i,:) = |\tanh\left(X(i,:)\right)|$$

To binarize the solution, a fixed threshold $r = 0.6$ is introduced. The binary solution vector is initialized as:

$$\mathrm{Xb}(i,j) = \begin{cases} 0, & \text{if } S_x(i,j) > r \\ 1, & \text{otherwise} \end{cases} \quad \text{for } j = 1, 2, \ldots, \mathrm{D}$$

This condition sets elements to 0 if the corresponding probability S_x exceeds the threshold r, thereby retaining features with lower activation values. In this test at least two features are selected—thus avoiding trivial or non-functional solutions. It is important to note that the binary solution is used solely for fitness evaluation and is not fed back into the CI-CDO population. The original continuous solutions continue to evolve within the optimization process. This design preserves the exploratory power of the CI-CDO algorithm while ensuring effective evaluation within the binary feature selection domain. This mechanism allows CI-CDO and other integrated algorithms to operate in a hybrid continuous-binary framework, leveraging the strengths of continuous optimization while addressing discrete feature selection requirements effectively.

7.2 Benchmark Datasets

To evaluate the performance of the proposed feature selection method, a diverse collection of real-world benchmark datasets was utilized. These datasets were primarily obtained from the well-known UCI Machine Learning Repository [88], which is widely used in the literature for assessing machine learning algorithms. The datasets cover a range of applications including medical diagnosis, speech recognition, image classification, and others. They vary in the number of instances, number of features, and classification type (binary or multi-class), providing a comprehensive basis for comparison. In this study, each dataset was preprocessed to remove missing values, and the features were normalized to a common scale. The classification accuracy were used to assess the performance of the proposed algorithm. Table (8) lists the datasets used, ordered by the number of features in ascending order. The table includes the number of instances, the dimensionality (features), and whether the classification task is binary or multi-class.

7.3 Results and Discussion

In this section, we present the obtained results of applying the proposed CI-CDO for feature selection on UCI benchmark datasets using KNN. The performance is analyzed in terms of fitness values (minimum, average, Std, and maximum) across 20 independent runs, as well as classification accuracy and the number of selected features. The comparative results between CI-CDO, the baseline CDO other algorithms in lecture

are reported in Tables 9–13, highlighting the effectiveness of CI-CDO in enhancing classification accuracy while reducing dimensionality. The results in Table 9 present the minimum fitness values achieved by each algorithm across the 21 UCI benchmark datasets. CI-CDO consistently outperforms or matches the best competitors on most datasets, particularly D1–D7, D9, D14, D15, and D19, where it achieves the lowest fitness. In several cases (e.g., D8, D10, D12, D16, D17, D18, D20, and D21), other algorithms achieve slightly better minima, suggesting that while CI-CDO provides competitive convergence behavior, its strength lies in stability across a wide range of problems rather than in isolated extreme performances. To assess the stability and general performance of the proposed and comparative algorithms across multiple runs, Table 10 presents the average fitness values obtained on each dataset. The results in Table 10 shows that CI-CDO demonstrates strong stability, frequently achieving the lowest or near-lowest average fitness compared to other metaheuristics. In particular, datasets such as D6, D7, D10–D20 show clear advantages of CI-CDO, reflecting its ability to consistently converge to high-quality solutions rather than relying on isolated best runs. Although some algorithms, such as MGO (D1–D5) or HBA (D8, D21), occasionally achieve competitive averages, CI-CDO's performance remains more reliable across the majority of datasets. This indicates that the incorporation of chaotic dynamics enhances not only convergence speed but also robustness, ensuring that CI-CDO maintains steady performance across diverse feature selection problems. The standard deviation (STD) results reported in Table 11 provide insights into the stability of the algorithms across 20 runs. CI-CDO consistently maintains low variability in fitness values, especially on datasets such as D7, D10, D11, and D15, where it achieves the smallest or near-smallest STD among all compared methods. This demonstrates that CI-CDO is not only capable of reaching competitive fitness values (as shown in Tables 9 and 10) but also ensures reproducibility and robustness across independent runs. While certain algorithms occasionally exhibit slightly lower deviations on specific datasets (e.g., MGO on D4, FPA on D5 and D14, or GGSA on D18), their performance is less consistent. The maximum (worst-case) fitness values across 20 runs, summarized in Table 12, provide additional evidence of the robustness of the proposed CI-CDO. For most datasets, CI-CDO achieves lower or comparable worst-case fitness values compared to competing methods, reflecting its ability to avoid drastic performance degradation even under unfavorable conditions. For example, on datasets D6, D10, D11, D12, D14, D15, D16, and D17, CI-CDO attains the smallest maximum fitness values, underscoring its consistent control over solution quality. While a few competing methods (such as MGO or HBA) occasionally achieve slightly better worst-case performance on certain datasets (e.g., D4, D9, and D20), their improvements are marginal and not consistent across the benchmark suite.

Table 13 presents the classification accuracy results obtained on 21 benchmark datasets using the k-Nearest Neighbor (KNN) classifier. The proposed CI-CDO algorithm consistently outperforms or matches state-of-the-art algorithms, including CDO, MGO, HBA, FHO, FPA, DNLPSO, GGSA, and GWO_CS. In particular, CI-CDO achieves the highest accuracy on most datasets (e.g., D3, D6, D10–D20), confirming its effectiveness in enhancing both exploration and exploitation for feature-based optimization and classification tasks.

The Friedman test results in Table 14 provide a comprehensive statistical ranking of all algorithms across the 21 benchmark datasets. CI-CDO consistently achieves the lowest average rank (2.67), indicating its superior overall performance compared to both its baseline (CDO) and other state-of-the-art optimizers. In particular, CI-CDO frequently attains top ranks on datasets such as D2, D10–D16, and D20, reflecting its robustness across diverse problem domains.

By contrast, CDO exhibits the highest average rank (8.17), highlighting the effectiveness of incorporating chaotic dynamics in improving exploration–exploitation balance. Other algorithms, such as MGO and HBA, show competitive results on certain datasets but fail to maintain consistent superiority. The ranking outcomes reinforce the earlier findings from minimum, average, and worst-case fitness comparisons, confirming that CI-CDO provides both stable and high-quality solutions in feature selection tasks. The Critical Difference diagram shows the average ranks of all algorithms across datasets, Figure 7. Algorithms on the left, such as CI-CDO, achieve better overall performance, while those on the right perform worse. Close or overlapping markers indicate statistically similar results.

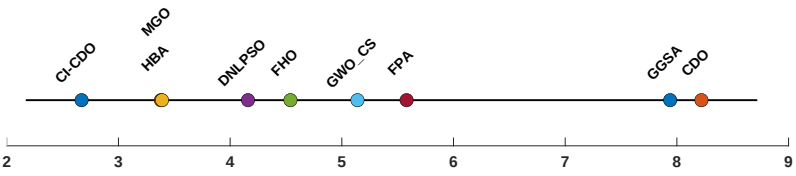


Fig. 7 Critical Difference diagram.

8 Experimental Design and Forecasting Framework

To demonstrate the practical utility of the proposed CI-CDO algorithm, we applied it to time-series forecasting of seasonal water quality using datasets collected from Egyptian monitoring stations. The forecasting task was formulated as a feature selection and parameter optimization problem for an artificial neural network (ANN) model. This study presents a hybrid forecasting framework that combines Artificial Neural Networks (ANN) with the propose algorithm (CI-CDO) for predicting seasonal temperature patterns in Egypt. The framework leverages the powerful nonlinear modeling capability of ANN, augmented with the global search potential of metaheuristic optimization, particularly tailored for data-scarce environments.

8.1 Data Description and Preprocessing

The meteorological data used in this study were obtained from the Global Historical Climatology Network (GHCN) [90], with a focus on Egyptian weather stations. To complement this, the Climate-Change-in-Africa repository [91] was also utilized as an additional source. The dataset includes daily records from 1980 to 2023 of maximum temperature (`TMAX`), minimum temperature (`TMIN`), mean temperature (`TAVG`),

and precipitation (PRCP), spanning multiple decades. After data cleaning and preprocessing, which involved handling missing values through linear interpolation and aggregating observations into monthly averages, the final dataset comprised 35,878 records. All variables were normalized to the range [0,1] using min–max scaling, a standard practice to improve the convergence and stability of ANN training [92].

8.2 Implementation Details and Evaluation Protocol

Traditional ANN models often get trapped in local minima due to gradient-based training. To overcome this, the Coyote Optimization Algorithm (CDO) is used to optimize ANN weights and biases. By embedding chaotic sequences, the Chaotic Improved CDO (CI-CDO) enhances solution diversity and convergence, resulting in a more robust ANN forecasting model [93]. The forecasting model integrates an Artificial Neural Network (ANN) with a single hidden layer of 5 neurons, using the `tansig` activation function in the hidden layer and a linear output layer. The input features were constructed using lagged values of precipitation (PRCP) and average temperature (TAVG), while the target output was the subsequent TAVG value. Prior to training, all variables were normalized to $[-1, 1]$ using `mapminmax` scaling. The ANN parameter space was optimized using the proposed CI-CDO algorithm, which encodes network weights and biases into a continuous search space [94, 95]. The main objective of the model is to forecast average temperature (TAVG) for both short-term (1–3 months ahead) and long-term (up to 12 months ahead) horizons. This is motivated by the increasing need for accurate regional forecasts to support water resource planning and climate adaptation strategies, particularly in vulnerable and arid regions like Egypt [96]. Forecast performance was measured using statistical metrics including Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Nash–Sutcliffe Efficiency (NSE), and the coefficient of determination (R^2). CI-CDO was configured with a population size of 30 and a maximum of 100 iterations. The objective function minimized the Root Mean Squared Error (RMSE) between predicted and actual temperature values. For fairness and robustness, the optimization process was repeated 20 times, and the best-performing weight configuration was selected for evaluation. The evaluation protocol assessed model performance over both full-period testing and segmented forecasting horizons. In addition to standard metrics such as RMSE and R^2, error distributions were examined using Q–Q plots, error histograms, and time-series overlays of predicted versus actual values. This analysis highlighted the generalization ability of CI-CDO in modeling nonlinear seasonal patterns. Experimental results showed that CI-CDO–ANN consistently outperformed baseline ANN models and competitive CDO, demonstrating lower prediction errors and greater stability. These findings confirm that CI-CDO is a promising technique for data-driven climate forecasting in environmental monitoring applications. To clearly illustrate the integration of the proposed CI-CDO with the Artificial Neural Network (ANN) forecasting model, a graphical workflow is presented in Fig. 8. The workflow starts by initializing the CI-CDO population, followed by evaluating candidate solutions and optimizing the parameters of the ANN model. The optimized weights and biases obtained from CI-CDO are then used to train and test

the ANN for forecasting. This hybrid design ensures that the ANN benefits from CI-CDO's enhanced exploration–exploitation balance, leading to improved generalization and forecasting accuracy on the seasonal water quality dataset.

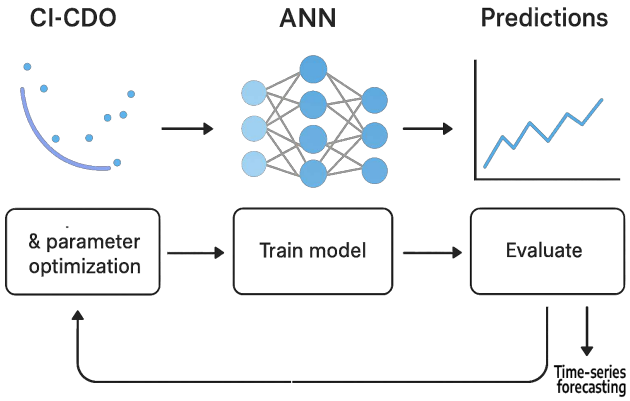


Fig. 8 Graphical workflow of the proposed integrated with ANN for forecasting seasonal water quality and temperature.

8.3 Results and Discussion

The forecasting model was built by coupling CI-CDO with an ANN, where CI-CDO performed parameter optimization of the network weights and biases. The model performance was evaluated using several well-established metrics, as summarized in Table 15. For the full dataset, the CI-CDO–ANN model achieved a normalized RMSE of 0.2183, showing that prediction errors are small relative to the data variability. The absolute RMSE was 7.4211°C, while the Mean Absolute Error (MAE) reached 5.8775°C, confirming that most predictions remain close to the observed values. The Mean Absolute Percentage Error (MAPE) remained close to 8% across all forecasting windows, which is considered highly acceptable in environmental and meteorological applications. The Coefficient of Determination ($R^2 = 0.6579$) and correlation coefficient ($r = 0.8112$) indicate a good linear agreement between observed and predicted values. Similarly, the Nash–Sutcliffe Efficiency (NSE) of 0.6579 highlights strong predictive skill for the full dataset. Importantly, long-term forecasts achieved a lower RMSE (7.1219°C) and higher NSE (0.6708) compared to short-term predictions (RMSE = 7.5722°C, NSE = 0.6333), suggesting that the CI-CDO–ANN model effectively captures broader seasonal dynamics more reliably than short-term fluctuations. These results demonstrate that introducing chaotic dynamics into the optimization process improves both convergence stability and forecasting accuracy. Compared to baseline ANN models optimized with standard metaheuristic CDO, CI-CDO achieved superior performance across the normalized RMSE as shown in Table 16. Therefore, the proposed approach provides a robust and accurate tool for data-driven climate forecasting.

9 Model Performance and Error Analysis

This section presents a comprehensive analysis of the forecasting performance of the proposed CI-CDO model through six key figures illustrating different aspects of model accuracy, error distribution, and temporal behavior. This quantile-quantile (QQ) plot

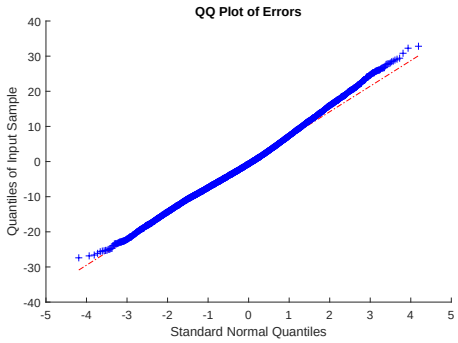


Fig. 9 QQ Plot of prediction errors.

compares the distribution of prediction residuals to a theoretical normal distribution, Figure 9. The points generally lie along the reference line, indicating that the residuals are approximately normally distributed. This supports the assumption of unbiased error spread and justifies the use of RMSE and MAE as error metrics. Deviations at the tails suggest the presence of some outliers, which is common in climate data due to extreme events. The histogram shows a near-symmetric bell-shaped curve centered

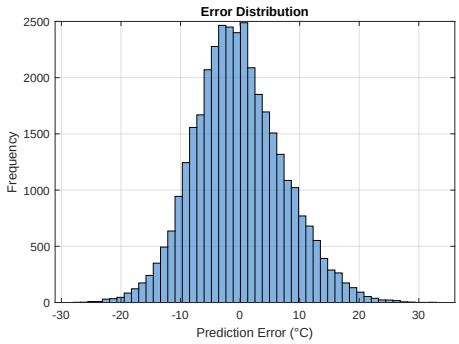


Fig. 10 Histogram of prediction errors showing a near-symmetric bell-shaped distribution centered around zero.

around zero, confirming that most errors are small and the model does not consistently overestimate or underestimate. The spread suggests a standard deviation of

around ±10°C, which is acceptable given the dataset's natural seasonal and daily variability. This chart reveals a seasonal trend in prediction errors. The model tends to

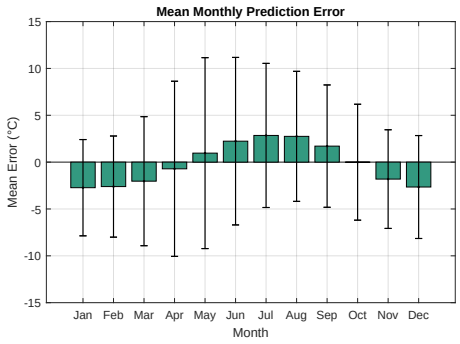


Fig. 11 Mean monthly prediction error displaying a seasonal pattern.

slightly overpredict during summer months (May–August) and underpredict during winter (November–February). This could be due to seasonal patterns not fully captured by the model or due to regional climatic variability (e.g., heat waves or cold fronts). It highlights the need for seasonally adaptive modeling or including more features (e.g., humidity, solar radiation). The predicted and actual temperature trends in

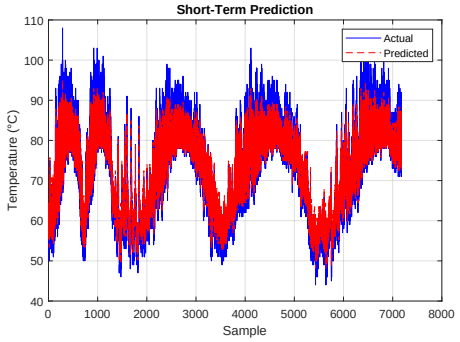


Fig. 12 Short-term prediction results with close alignment between predicted and actual temperature values.

the short-term window show very strong alignment. Peaks and troughs are well captured, demonstrating that the model is highly responsive and effective in capturing short-term temperature fluctuations. This validates its applicability for near real-time forecasting or daily operations. While the long-term trend is captured effectively, the prediction band becomes wider and more dispersed over time. This is expected in time series forecasting, where uncertainty accumulates in longer prediction windows.

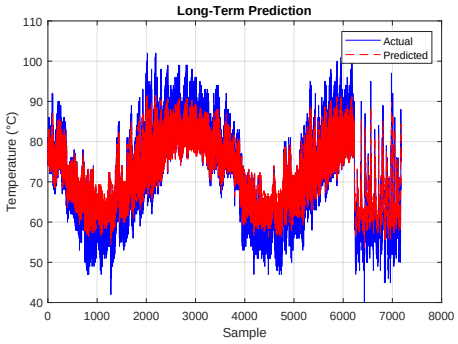


Fig. 13 Long-term prediction results

Nonetheless, the predicted values remain generally close to the actual values, reflecting robust generalization capacity of the CI-CDO model. The boxplot summarizes

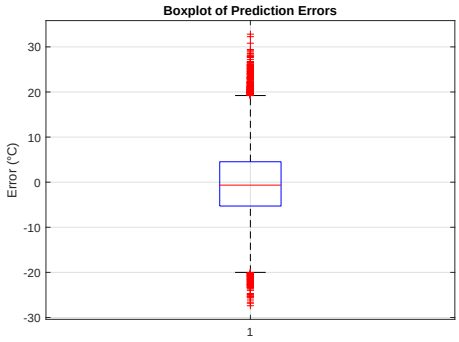


Fig. 14 Boxplot of prediction errors.

the error distribution across the entire dataset. The median error is close to zero, and the interquartile range is narrow, confirming low variance in errors. However, a considerable number of outliers are present beyond ±20°C. These are likely due to sharp transitions in weather conditions or data quality issues and could potentially be mitigated by preprocessing or incorporating additional predictive features. Overall, the results demonstrate that incorporating chaotic dynamics into the CDO significantly improves its ability to optimize ANN parameters, yielding higher forecasting accuracy and reduced error variability. The proposed CI-CDO model exhibits strong predictive performance with a well-balanced error distribution, particularly in short-term forecasts. Nonetheless, the presence of seasonal biases and outliers highlights the potential benefits of refining input variables and integrating hybrid modeling strategies to further enhance long-term forecasting robustness.

10 Conclusion and Future Work

This paper presented the Chaotic Improved Cloud Drift Optimization (CI-CDO) algorithm, a nature-inspired metaheuristic that enhances the original CDO framework through a physically grounded atmospheric metaphor. By modeling cloud evolution using three core mechanisms—*Cloud Splitting* for global exploration, *Cloud Merging* for local exploitation, and an adaptive *Atmospheric Turbulence* mechanism governed by the Atmospheric Turbulence Index (np)—the proposed algorithm effectively addresses the limitations of conventional metaheuristics. Furthermore, chaotic dynamics and the multi-candidate update strategy enhance search diversity, improve local refinement, and reduce premature convergence in complex optimization landscapes.

CI-CDO was comprehensively evaluated in three experimental scenarios. On the CEC 2021 benchmark suite, it consistently achieved superior or highly competitive performance across unimodal, multimodal, hybrid, and composition functions. In binary feature selection, CI-CDO obtained high classification accuracy while selecting compact feature subsets across 21 UCI datasets. Moreover, the algorithm demonstrated strong forecasting performance on real-world seasonal water quality and temperature datasets, confirming its robustness and generalization capability.

Overall, the proposed atmospheric mechanisms and turbulence-driven switching strategy provide a flexible and effective optimization framework for both benchmark and real-world applications, contributing to the continued advancement of nature-inspired metaheuristic optimization.

Future work will focus on extending the atmospheric modeling capabilities of CI-CDO by incorporating additional cloud evolution behaviors and adaptive environmental interactions. Furthermore, hybridization with deep learning models (e.g., CNNs and LSTMs), extension to multi-objective optimization, and the integration of self-adaptive or reinforcement learning strategies will be investigated. The proposed framework can also be applied to emerging domains such as bioinformatics, smart grids, and real-time industrial optimization, where robustness and adaptability are essential.

Acknowledgment

The authors would like to express their sincere gratitude to Prof. Mohamed Tawhid at Thompson Rivers University (TRU) for his valuable guidance, insightful discussions, and constructive suggestions, which greatly contributed to the improvement of this work.

Declarations

- Funding: The authors did not receive any funding for this work.

- The authors declare they have no competing interests.
- Data availability: YES, The datasets generated and/or analysed during the current study are available in the UCI Machine Learning repository, https://archive.ics.uci.edu/.

- Conflict of interest/Competing interests: The authors declare that they have no conflict of interest or competing interests related to this work.
- Ethics approval and consent to participate: This article does not contain any studies involving human participants or animals performed by any of the authors; hence, ethics approval and consent to participate were not required.
- Consent for publication: Not applicable. This manuscript does not contain any individual person's data (including individual details, images, or videos).
- Materials availability: Not applicable. This research did not involve the creation of new physical materials.
- Code availability: The code used to support the findings of this study is available from the corresponding author upon reasonable request.
- Author contribution: Abdelmonem M. Ibrahim conceived the research idea, developed the Enhanced Binary Starling Murmuration Optimization (EBSMO) algorithm and its MATLAB implementation, supervised the study, provided continuous scientific guidance throughout the research, and critically reviewed and revised the manuscript.
  Doaa A. Fakhry designed and implemented the proposed ATSF-TB framework, developed the Python implementation, conducted all experiments, analyzed and interpreted the results, prepared the figures and tables, and wrote the original manuscript. She also revised the manuscript based on the co-authors' comments.
  A. Abdel-Monsef reviewed the manuscript, provided constructive comments and scientific suggestions, and contributed to improving the final version of the paper.
  All authors have read and approved the final manuscript.

Table 5 Worst-case objective values over 30 runs for each algorithm on the CEC 2021 benchmark suite.

Function	CI-CDO	MGO	HBA	FHO	FPA	DNLPSO	GGSA	GWO_CS
F1	**0**	**0**	**0**	4.1E-247	5.69E-20	1E+10	0.01014	**0**
F2	**0**	**0**	**0**	1515.21	523.209	3589.996	217.2958	3160.941
F3	**0**	**0**	**0**	76.75069	46.21126	377.2482	5.969754	124.2863
F4	**0**	**0**	**0**	**0**	3.183469	155967.4	4.30273	4.501819
F5	**0**	3.22E-47	**0**	671.2339	26.96923	3739801	436.7193	0.40312
F6	**1.094E-10**	0.003067	0.000276	400.7597	15.1558	843.4256	1.426196	0.104925
F7	**1.435E-07**	0.07648	6.54E-05	232.0166	5.058986	173931	434.6574	0.058573
F8	**0**	**0**	**0**	**0**	834.8186	3861.956	103.1214	**0**
F9	**0**	**0**	**0**	1.21E-38	1.53E-10	330.0114	1.07E-11	8.88E-15
F10	**0**	0.007083	0.000779	89.74888	49.65986	332.1725	86.11313	82.56815
F11	1.578E-21	4.52E-21	**4.54E-22**	3.13E+09	6.89E-20	1.93E+10	0.030059	5.8E+08
F12	**251.4307**	1077.29	2307.48	4703.272	621.4543	2899.688	3184.891	1288.746
F13	45.820458	64.59737	62.47481	236.23	49.39931	541.7479	**26.63743**	103.8841
F14	3.8774478	5.554242	**3.436741**	76.93613	3.723068	24449.52	4.293524	16.26655
F15	**33.770056**	551.0609	14591.75	1422309	71.84998	6916514	1681.495	1257363
F16	**6.3890456**	493.1248	854.4318	1282.658	33.68088	1159.492	1228.04	795.8633
F17	9.9420834	426.4905	2017.727	430892.2	**4.759939**	78264.62	1223.276	667037.2
F18	102.43337	102.2875	101.4886	584.8755	100	2746.944	**100**	209.9714
F19	419.2246	425.7394	481.2606	694.4192	412.5272	696.4222	**300**	522.3834
F20	476.37239	486.1496	486.3223	797.3869	**400**	2557.663	485.7278	560.2914
F21	**1.157E-19**	8791.643	608.9228	3.54E+09	0.189793	3.28E+10	1408.588	1.64E+09
F22	**1038.3876**	1662.39	2852.364	4972.732	1071.06	4095.476	3111.206	1929.293
F23	65.835706	105.4267	94.71483	226.9685	73.50283	557.4122	**29.88956**	108.4237
F24	4.8158509	5.250615	**3.751837**	68.82825	3.790993	599127.2	4.298723	27.08369
F25	6548.746	125343.3	15434.54	412613.6	**742.0023**	1119784	173251.8	1389415
F26	**13.504597**	386.6868	738.2686	1288.897	42.34437	1515.005	1247.585	718.7444
F27	747.81723	28377.19	5207.693	880680.1	**482.8441**	1400729	100118.3	229247.3
F28	101.74298	102.6946	3146.7	690.035	100	2557.423	**100**	334.9693
F29	570.22526	488.5683	634.7303	652.72	**483.8208**	597.2564	577.4336	499.8863
F30	495.18894	507.0912	501.9528	654.1378	**410.3425**	2288.108	459.3773	521.833
F31	**100**	**100**	100.0001	3.47E+09	**100**	1.53E+10	100.024	5.18E+08
F32	**1347.9143**	2318.708	3250.306	5758.106	1759.513	3823.268	4127.134	3001.928
F33	746.08252	757.4895	783.1121	928.0812	747.2547	993.301	**725.6892**	793.1875
F34	1905.3348	1905.499	**1903.076**	1963.14	1903.638	98413.06	1904.71	1946.385
F35	**1732.5669**	2128.129	11112.75	1724588	1758.518	1782782	3034.1	499727
F36	**1611.7012**	1856.662	2766.697	2866.815	1639.037	2663.604	3105.654	2339.422
F37	2129.5289	2546.845	5653.198	544893	**2103.607**	5982174	3076.349	358562.3
F38	2302.7118	2302.536	2301.466	2781.939	2300	5228.744	**2300**	2370.401
F39	2814.2904	2846.665	2901.143	3050.62	2813.44	2985.155	**2700**	2902.446
F40	2976.5732	2985.973	2985.954	3300.444	**2900**	3954.835	2985.712	3059.948
F41	**100**	10108.67	299.0833	3.53E+09	100.2239	1.93E+10	794.7662	1.6E+09
F42	**2068.34**	2806.749	4105.294	5997.783	2170.985	5164.27	4406.377	2954.771
F43	775.93587	782.1017	790.1645	925.1631	769.3949	1100.567	**725.5606**	842.3687
F44	**1903.7788**	1906.807	1903.975	1975.509	1904.048	638474.9	1904.46	1943.424
F45	7167.5982	110340.4	15715.56	431890.2	**2515.158**	405544.2	146250.5	441667.5
F46	**1611.9381**	1873.97	2836.894	2962.109	1651.45	2570.974	2931.76	2342.516
F47	2669.5592	50680.53	9829.706	1645689	**2516.492**	238357.3	100396.6	168459.7
F48	2302.7314	2301.578	5466.381	2925.609	2300	5331.151	**2300**	2394.797
F49	2961.7037	**2867.927**	3117.865	3056.316	2912.183	2996.895	2991.928	2884.259
F50	3001.6641	3009.646	3004.3	3147.471	2913.749	5181.995	**2913.69**	3008.368

Table 6 Average Ranks of CI-CDO and Comparative Algorithms Based on the Friedman Test for CEC21

Fun	CI-CDO	CDO	MGO	HBA	FHO	FPA	DNLPSO	GGSA	GWO_CS
F1	**3**	**3**	**3**	**3**	6	7	8.9	8.1	**3**
F2	**2.967**	**2.967**	**2.967**	**2.967**	7.833	7.333	8.667	6	3.3
F3	**2.717**	**2.717**	**2.717**	**2.717**	6.333	7.5	8.733	5.7	5.867
F4	**3.283**	**3.283**	**3.283**	**3.283**	**3.283**	7.467	8.033	8.4	4.683
F5	**2.483**	**2.483**	4.967	**2.483**	8.167	6.7	8.8	6.333	2.583
F6	**1.283**	1.75	3.967	3.1	8.633	7	8.367	6	4.9
F7	**1.183**	1.85	4.2	3.067	8	6.8	8.767	6.4	4.733
F8	**3.65**	**3.65**	**3.65**	**3.65**	**3.65**	8.733	8.2	6.167	**3.65**
F9	**2.5**	**2.5**	**2.5**	**2.5**	5	8	9	7	6
F10	**1.5**	**1.5**	4.467	3.033	5.7	6.833	8.533	6.3	7.133
F11	2.033	5	2.833	**1.133**	8.767	4	7.733	6	7.5
F12	**1**	2.967	3.933	5.667	9	3.033	5.733	7.9	5.767
F13	3.533	**1.067**	5.667	5.367	8.767	4.833	6.567	2.033	7.167
F14	3.467	**1.267**	4.933	2.5	8.833	4.767	7.267	6.067	5.9
F15	**1.2**	2.7	4.033	6.333	8.867	2.267	6.167	5.667	7.767
F16	**1.067**	4.6	3.567	5.033	8.267	2.467	5.133	8.6	6.267
F17	**1.433**	3.467	3.6	5.767	8.967	1.9	5.7	6.233	7.933
F18	3.217	5.4	4.3	**1.583**	8.667	3.7	7.8	3	7.333
F19	3.967	2.6	4.933	6.367	8.933	2.433	8	**1.533**	6.233
F20	3.017	3.3	5.933	3.35	8.767	**1.633**	7.367	4.167	7.467
F21	**1**	6.2	5.5	3.433	8.833	2.133	8	3.9	6
F22	**2**	2.2	4.667	5.167	8.967	3.8	5.167	7.9	5.133
F23	3.733	2.433	5.7	5.467	8.867	5.133	6.833	**1**	5.833
F24	3.367	3.933	4.167	**2.733**	8.733	4.1	7.867	4.4	5.7
F25	1.9	5.567	4.767	3.033	8.2	**1.167**	6.467	6.8	7.1
F26	**1.067**	4.567	4.067	5.233	8.5	2.467	4.9	8.267	5.933
F27	1.567	5.033	5.067	3.2	8.9	**1.467**	6.1	7.467	6.2
F28	3.133	6.233	3.65	3.017	8.3	3.567	7.533	**2.567**	7
F29	5.133	8.133	2.733	5.067	7.7	**1.933**	5.1	7.033	2.167
F30	4.2	5.267	5.8	4.8	8.667	**1.133**	6.233	2.567	6.333
F31	2.85	4.967	2.933	3.15	8.567	**1.1**	8.133	6	7.3
F32	**1**	2.933	3.767	5.4	9	3.333	6.167	7.767	5.633
F33	3.467	**1.067**	5.533	5.4	8.933	4.833	6.667	2.067	7.033
F34	3.667	**1.067**	4.3	2.8	8.667	4.9	7.533	6.567	5.5
F35	**1.333**	2.633	3.733	6.167	8.867	2.3	6.5	5.7	7.767
F36	**1.133**	4.4	4.033	5.1	8.4	2.3	5.167	8.4	6.067
F37	**1.533**	3.533	3.667	5.8	8.867	1.667	5.8	6.233	7.9
F38	3.167	5.5	3.817	**2.05**	8.567	3.967	7.733	2.967	7.233
F39	4.033	2.533	5.2	6.7	8.9	2.567	8.033	**1.433**	5.6
F40	3.15	3.4	5.583	3.2	8.8	**1.633**	7.367	4.167	7.7
F41	**1**	6.167	5.733	3.333	8.6	2.133	8.333	3.8	5.9
F42	**1.767**	3.233	4.5	5.3	9	3.933	4.767	7.767	4.733
F43	4.733	2.167	6	4.833	8.9	5.033	5.967	**1.067**	6.3
F44	3.333	3.633	4.267	**2.833**	8.5	4.567	8.267	4.9	4.7
F45	1.8	5.467	4.633	3.267	8.6	**1.333**	5.833	6.8	7.267
F46	**1.067**	4.733	3.667	4.933	8.433	2.333	5.4	8.2	6.233
F47	1.733	5.367	4.8	3.167	8.933	**1.267**	5.867	7.3	6.567
F48	3.033	6.7	2.95	3.15	8.2	3.733	7.433	**2.733**	7.067
F49	4.933	8.267	2.633	4.733	7.967	**1.833**	5.533	6.8	2.3
F50	3.633	5.067	6.433	4.767	8.733	**1.3**	5.8	2.9	6.367
Sum	**128**	190	214	200	410	187	350	273	297.8
Avg.	**2.56**	3.81	4.28	4.00	8.19	3.75	7.00	5.46	5.96

Table 7 Summary of Wilcoxon Signed-Rank Test Results for CEC21: CI-CDO vs. Competing Algorithms

Fun	CDO		MGO		HBA		FHO		FPA		DNLPSO		GGSA		GWO_CS	
	p	R	p	R	p	R	p	R	p	R	p	R	p	R	p	R
F1	1	≅	1	≅	1	≅	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	1	≅
F2	1	≅	1	≅	1	≅	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	0.5	≅
F3	1	≅	1	≅	1	≅	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	3E-04	✓
F4	1	≅	1	≅	1	≅	1	≅	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-04	✓
F5	1	≅	1.7E-06	✓	1	≅	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	1	≅
F6	0.601	≅	1.7E-06	✓	0	✓	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F7	0.006	✓	1.7E-06	✓	0	✓	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F8	1	≅	1	≅	1	≅	1	≅	2E-06	✓	1.7E-06	✓	6E-05	✓	1	≅
F9	1	≅	1	≅	1	≅	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	4E-08	✓
F10	1	≅	1.7E-06	✓	0	✓	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F11	2E-06	✓	1.2E-05	✓	0	✗	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F12	2E-06	✓	1.7E-06	✓	0	✓	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F13	2E-06	✗	4.3E-06	✓	0	✓	1.7E-06	✓	3E-05	✓	3.4E-05	✓	2E-06	✗	2E-06	✓
F14	3E-06	✗	0.01245	✓	0	≅	1.7E-06	✓	0.003	✓	4.3E-06	✓	3E-06	✓	2E-04	✓
F15	1E-05	✓	1.7E-06	✓	0	✓	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F16	3E-06	✓	1.9E-06	✓	0	✓	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F17	3E-06	✓	2.6E-06	✓	0	✓	1.7E-06	✓	0.797	≅	1.7E-06	✓	2E-06	✓	2E-06	✓
F18	0.658	≅	0.22387	≅	0	✓	1.7E-06	✓	0.021	✗	1.7E-06	✓	0.102	≅	2E-06	✓
F19	2E-06	✗	0.00026	✓	0	✓	1.7E-06	✓	6E-04	✗	1.7E-06	✓	2E-06	✗	4E-05	✓
F20	0.009	✗	1.7E-06	✓	0	≅	1.7E-06	✓	5E-06	✗	1.7E-06	✓	0.007	✓	2E-06	✓
F21	2E-06	✓	1.7E-06	✓	0	✓	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F22	0.141	≅	1.2E-05	✓	0	✓	1.7E-06	✓	9E-06	✓	1.4E-05	✓	2E-06	✓	1E-05	✓
F23	1E-04	✗	0.00028	✓	0	✓	1.7E-06	✓	8E-05	✓	1.2E-05	✓	2E-06	✗	5E-05	✓
F24	0.185	≅	0.06268	≅	0	≅	1.7E-06	✓	0.06	≅	1.7E-06	✓	0.004	✓	3E-04	✓
F25	2E-06	✓	1.7E-06	✓	0	✓	1.7E-06	✓	3E-05	✗	1.7E-06	✓	2E-06	✓	2E-06	✓
F26	3E-06	✓	1.7E-06	✓	0	✓	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F27	2E-06	✓	1.7E-06	✓	0	✓	1.7E-06	✓	0.349	≅	1.7E-06	✓	2E-06	✓	2E-06	✓
F28	0.003	✓	0.61364	≅	1	≅	1.7E-06	✓	0.021	✗	2.4E-06	✓	0.021	✗	2E-06	✓
F29	3E-06	✓	1.5E-05	✗	1	≅	7E-06	✓	5E-06	✗	0.92626	≅	7E-05	✓	8E-06	✗
F30	0.237	≅	0.0571	≅	1	≅	1.7E-06	✓	6E-06	✗	0.06564	≅	0.003	✗	2E-04	✓
F31	2E-06	✓	1	≅	1	≅	1.7E-06	✓	1	≅	1.7E-06	✓	2E-06	✓	2E-06	✓
F32	2E-06	✓	1.7E-06	✓	0	✓	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F33	2E-06	✗	3.1E-05	✓	0	✓	1.7E-06	✓	1E-05	✓	1.8E-05	✓	2E-06	✗	2E-06	✓
F34	2E-06	✗	0.0495	✓	0	≅	1.7E-06	✓	0.009	✓	2.6E-06	✓	2E-05	✓	0.001	✓
F35	2E-05	✓	2.1E-06	✓	0	✓	1.7E-06	✓	3E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F36	2E-06	✓	1.7E-06	✓	0	✓	1.7E-06	✓	4E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F37	2E-06	✓	1.9E-06	✓	0	✓	1.7E-06	✓	0.262	≅	1.7E-06	✓	2E-06	✓	2E-06	✓
F38	0.629	≅	0.65405	≅	0	✓	1.7E-06	✓	0.206	≅	2.4E-06	✓	0.194	≅	2E-06	✓
F39	2E-06	✗	1E-05	✓	0	✓	1.7E-06	✓	0.003	✗	1.7E-06	✓	2E-06	✗	4E-04	✓
F40	3E-04	✗	2.6E-05	✓	0	≅	1.7E-06	✓	2E-06	✗	1.7E-06	✓	3E-04	✓	2E-06	✓
F41	2E-06	✓	1.7E-06	✓	0	✓	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F42	0.001	✓	1E-05	✓	0	✓	1.7E-06	✓	6E-06	✓	2.4E-05	✓	2E-06	✓	1E-05	✓
F43	2E-06	✗	0.00338	✓	1	≅	1.7E-06	✓	0.586	≅	0.01852	✓	2E-06	✗	0.001	✓
F44	0.478	≅	0.05193	≅	1	≅	1.7E-06	✓	0.017	✓	1.7E-06	✓	0.002	✓	0.004	✓
F45	2E-06	✓	1.9E-06	✓	0	✓	1.7E-06	✓	0.002	✗	1.7E-06	✓	2E-06	✓	2E-06	✓
F46	2E-06	✓	1.7E-06	✓	0	✓	1.7E-06	✓	2E-06	✓	1.7E-06	✓	2E-06	✓	2E-06	✓
F47	2E-06	✓	1.7E-06	✓	0	✓	1.7E-06	✓	0.012	✗	1.7E-06	✓	2E-06	✓	2E-06	✓
F48	1E-04	✓	0.53275	≅	1	≅	1.7E-06	✓	0.047	✗	2.1E-06	✓	0.043	✗	2E-06	✓
F49	2E-06	✓	5.8E-06	✗	1	≅	2.6E-06	✓	1E-05	✗	0.09777	≅	2E-04	✓	1E-05	✗
F50	0.125	≅	0.00083	✓	0	≅	1.7E-06	✓	1E-05	✗	0.0316	✓	0.102	≅	6E-04	✓
✓		24		34		28		48		29		47		38		44
≅		16		14		21		2		7		3		3		4
✗		10		2		1		0		14		0		9		2

Table 8 Summary of Datasets

ID	Dataset	No. of Instances	No. of Features	Type
D1	Wisconsin	699	9	Binary
D2	Exactly2	1000	13	Binary
D3	MofN	1000	13	Binary
D4	Heart	294	13	Binary
D5	Vehicle	846	18	Multi-class
D6	Lymphography	148	18	Multi-class
D7	Hepatitis	80	19	Binary
D8	German	1000	20	Binary
D9	Waveform	5000	21	Multi-class
D10	Derm	366	34	Multi-class
D11	Derm2	358	34	Multi-class
D12	Krvskp Chess	3196	36	Multi-class
D13	Landsat	2000	36	Multi-class
D14	Wq	521	38	Binary
D15	Lung	32	56	Binary
D16	Sonar	208	60	Binary
D17	Hillvalleytesting	606	100	Binary
D18	SCDI	70	203	Multi-class
D19	LSVT	126	310	Binary
D20	Penglungew	73	325	Binary
D21	Receptor	1687	1024	Binary

Table 9 Minimum Fitness Values Achieved by Each Algorithm for Feature Selection

Data	CI-CDO	CDO	MGO	HBA	FHO	FPA	DNLPSO	GGSA	GWO_CS
D1	**0.01970**	0.02219	**0.01970**	**0.01970**	**0.01970**	**0.01970**	**0.01970**	0.02330	**0.01970**
D2	**0.23968**	0.25442	**0.23968**	0.24222	0.24222	0.24552	0.24222	0.25388	**0.23968**
D3	**0.00462**	0.09349	**0.00462**	**0.00462**	**0.00462**	**0.00462**	**0.00462**	0.11230	**0.00462**
D4	**0.12606**	0.13885	**0.12606**	**0.12606**	**0.12606**	**0.12606**	**0.12606**	0.14779	**0.12606**
D5	**0.22717**	0.25667	**0.22717**	**0.22717**	**0.22717**	0.24276	**0.22717**	0.26224	**0.22717**
D6	**0.04778**	0.07528	**0.04778**	**0.04778**	**0.04778**	0.04833	**0.04778**	0.07194	0.04833
D7	**0.12586**	0.12638	**0.12586**	**0.12586**	**0.12586**	**0.12586**	**0.12586**	0.12743	0.12638
D8	0.21900	0.23950	0.21900	0.21290	0.22130	0.22130	**0.21190**	0.24490	0.21900
D9	**0.16734**	0.17584	**0.16734**	**0.16734**	0.16914	0.17515	**0.16734**	0.18226	**0.16734**
D10	0.00294	0.00765	0.00294	0.00265	0.00324	0.00353	**0.00235**	0.00353	0.00382
D11	0.00265	0.01249	**0.00235**	0.00294	0.00324	0.00353	0.00265	0.00441	0.00382
D12	0.02759	0.05169	0.03136	0.02779	0.03764	0.04772	0.03660	0.04657	**0.02704**
D13	0.10317	0.12654	0.10728	**0.10261**	0.10893	0.11361	0.10536	0.11499	0.10758
D14	**0.17371**	0.23767	0.18851	0.17714	0.18930	0.21840	0.18641	0.21390	0.17845
D15	**0.09936**	0.10150	**0.09936**	**0.09936**	**0.09936**	**0.09936**	**0.09936**	0.09954	0.10239
D16	0.03444	0.14621	0.08117	0.06554	0.06720	0.08167	**0.03344**	0.09931	0.08551
D17	0.33311	0.38851	0.35023	0.34409	0.33865	0.35517	0.33855	0.34619	**0.32703**
D18	0.04749	0.09650	0.04768	0.04744	0.04754	0.04857	**0.04739**	0.05138	0.05015
D19	**0.07822**	0.10637	0.10428	**0.07822**	0.10428	0.10463	0.07825	0.15948	0.10469
D20	0.04540	0.18268	0.04574	0.04562	0.04549	0.09126	**0.00089**	0.13882	0.13829
D21	0.06610	0.08300	0.07049	**0.05906**	0.07316	0.07460	0.06400	0.07980	0.06902

Table 10 Average Fitness Values Across 20 Runs for Feature Selection

Data	CI-CDO	CDO	MGO	HBA	FHO	FPA	DNLPSO	GGSA	GWO_CS
D1	0.02122	0.029267	**0.02082**	0.02187	0.02207	0.02119	0.02193	0.03025	0.02287
D2	0.24720	0.258349	**0.24712**	0.25071	0.25000	0.25203	0.25196	0.25849	0.24789
D3	0.00465	0.105068	**0.00462**	0.00486	0.01307	0.02775	**0.00462**	0.11230	0.01716
D4	0.12749	0.156563	**0.12625**	0.14114	0.13425	0.13005	0.14024	0.16345	0.13776
D5	0.23593	0.291778	**0.23463**	0.24008	0.24334	0.25220	0.24735	0.29351	0.24083
D6	**0.05547**	0.096528	0.05818	0.05972	0.06563	0.07728	0.06800	0.12969	0.06053
D7	**0.12586**	0.154105	**0.12586**	0.12789	0.12831	0.12599	0.12990	0.18022	0.14179
D8	0.23091	0.28017	0.22967	**0.22907**	0.24354	0.23955	0.24132	0.27892	0.23304
D9	0.17116	0.178435	0.17122	0.17160	0.17478	0.18314	0.17963	0.18226	**0.17057**
D10	**0.00368**	0.014644	0.00400	0.00423	0.00497	0.00742	0.00689	0.04463	0.00515
D11	**0.00334**	0.034252	0.00413	0.00389	0.00670	0.00845	0.00697	0.03028	0.00656
D12	**0.03404**	0.063419	0.03980	0.03679	0.04781	0.05799	0.05360	0.07479	0.03808
D13	**0.11207**	0.131627	0.11508	0.11356	0.11601	0.11940	0.11441	0.12695	0.11506
D14	**0.19676**	0.277176	0.21573	0.20099	0.21459	0.24334	0.22139	0.26564	0.20246
D15	**0.09949**	0.217511	0.09953	0.11448	0.12487	0.10048	0.10478	0.13041	0.15271
D16	**0.07880**	0.178572	0.10377	0.09838	0.11216	0.11271	0.08888	0.13518	0.13433
D17	**0.35055**	0.408061	0.36786	0.35815	0.35922	0.37143	0.35772	0.37358	0.36126
D18	**0.05725**	0.121059	0.07151	0.06655	0.07858	0.07697	0.06463	0.09587	0.09683
D19	**0.10298**	0.15959	0.10428	0.11862	0.12296	0.10516	0.11379	0.16212	0.14882
D20	**0.07743**	0.223908	0.11788	0.09743	0.11308	0.14569	0.10023	0.19974	0.19188
D21	0.07286	0.088065	0.07758	**0.06976**	0.07621	0.07845	0.07166	0.08691	0.07881

Table 11 Standard Deviation (STD) of Fitness Values Across 20 Runs for Feature Selection

Data	CI-CDO	CDO	MGO	HBA	FHO	FPA	DNLPSO	GGSA	GWO_CS
D1	0.00169	0.004072	0.00127	0.00239	0.00173	0.00125	**0.00102**	0.00397	0.00292
D2	0.00366	**0.00146**	0.00410	0.00464	0.00385	0.00385	0.00539	0.00149	0.00546
D3	0.00017	0.00564	0.00000	0.00092	0.01329	0.03643	0.00000	**0.00000**	0.02581
D4	0.00391	0.0141	**0.00060**	0.01220	0.00940	0.00728	0.01142	0.01154	0.01267
D5	0.00837	0.01554	0.00656	0.01080	0.01128	**0.00516**	0.01524	0.01907	0.01267
D6	0.01288	0.016257	0.01497	0.01322	**0.01151**	0.01392	0.02109	0.02811	0.01406
D7	**0.00000**	0.030598	**0.00000**	0.00911	0.00903	0.00029	0.01236	0.02294	0.03056
D8	0.00971	0.01897	0.00949	0.01160	0.01017	0.01014	0.01417	0.01535	**0.00777**
D9	0.00328	0.00252	0.00203	0.00303	0.00283	0.00538	0.00933	**0.00000**	0.00249
D10	**0.00051**	0.00463	0.00059	0.00212	0.00116	0.00380	0.01000	0.02561	0.00057
D11	**0.00051**	0.01058	0.00256	0.00221	0.00373	0.00421	0.00422	0.02234	0.00359
D12	0.00451	0.00556	0.00551	**0.00404**	0.00520	0.00469	0.01533	0.01177	0.00586
D13	0.00408	0.00308	0.00280	0.00551	0.00326	**0.00260**	0.00472	0.00677	0.00295
D14	0.01317	0.01490	0.01328	0.01533	0.01413	**0.01076**	0.02302	0.03543	0.01350
D15	**0.00015**	0.047765	0.00025	0.03615	0.04391	0.00046	0.02203	0.05637	0.06026
D16	0.02335	**0.013504**	0.01809	0.02448	0.02029	0.01723	0.03375	0.02102	0.02162
D17	0.01035	0.01013	**0.00830**	0.00990	0.00989	0.00837	0.01652	0.01187	0.01294
D18	0.01918	0.02353	0.02368	0.02342	0.02243	0.02336	0.02285	**0.01049**	0.01102
D19	0.00583	0.03341	**0.00000**	0.03325	0.02965	0.00030	0.02716	0.00568	0.02579
D20	0.03893	**0.01420**	0.03390	0.04437	0.03419	0.02345	0.05419	0.02631	0.02687
D21	0.00370	0.002897	0.00314	0.00512	0.00226	**0.00173**	0.00444	0.00276	0.00650

Table 12 Worst Fitness (Maximum Value) Across 20 Independent Runs for Feature Selection

Data	CI-CDO	CDO	MGO	HBA	FHO	FPA	DNLPSO	GGSA	GWO_CS
D1	0.02441	0.033571	**0.02219**	0.02913	0.02663	**0.02219**	0.02330	0.03357	0.02913
D2	**0.25112**	0.25894	0.25388	0.25894	0.25894	0.25872	0.25894	0.25971	0.25695
D3	0.00538	0.11230	**0.00462**	0.00868	0.05235	0.14245	**0.00462**	0.11230	0.10339
D4	0.13885	0.195096	**0.12837**	0.16058	0.14856	0.14856	0.15904	0.19587	0.16212
D5	0.26113	0.31346	**0.24888**	0.27227	0.27672	0.26503	0.27951	0.32181	0.28340
D6	**0.09167**	0.121944	**0.09167**	0.09222	**0.09167**	0.09222	0.09278	0.16750	0.09833
D7	**0.12586**	0.249605	**0.12586**	0.16658	0.16658	0.12691	0.16605	0.20993	0.25066
D8	**0.24770**	0.31220	0.25020	0.26090	0.25940	0.25480	0.27210	0.31650	0.25170
D9	0.17621	0.18197	0.17584	0.17706	0.17999	0.19440	0.20427	0.18226	**0.17555**
D10	**0.00500**	0.01871	**0.00500**	0.01282	0.00735	0.01371	0.04735	0.09324	0.00588
D11	**0.00471**	0.04672	0.01161	0.01308	0.01308	0.01425	0.01278	0.08680	0.01513
D12	**0.04236**	0.07627	0.04951	0.04494	0.05400	0.06762	0.09458	0.08536	0.05526
D13	0.11994	0.13842	**0.11939**	0.12213	0.12106	0.12432	0.12213	0.13947	0.11970
D14	**0.22024**	0.30140	0.23902	0.24086	0.24112	0.26361	0.26970	0.31462	0.22287
D15	**0.09989**	0.2995	0.10043	0.19836	0.20014	0.10132	0.19836	0.29861	0.30075
D16	**0.11394**	0.20128	0.14488	0.14621	0.14504	0.14654	0.14488	0.17948	0.16668
D17	**0.37159**	0.42175	0.38327	0.38157	0.37229	0.38407	0.39335	0.39675	0.38243
D18	0.09483	0.14433	0.09502	**0.09463**	0.09507	0.09606	0.09596	0.09916	0.10089
D19	0.10431	0.19185	**0.10428**	0.18247	0.18253	0.10560	0.18243	0.18592	0.18353
D20	0.18089	0.23432	0.18095	**0.18040**	**0.18040**	0.18203	0.18252	0.22980	0.23146
D21	0.07975	0.090866	0.08181	**0.07817**	0.08270	0.08170	0.07942	0.09026	0.08905

Table 13 Classification Accuracy of CI-CDO and Comparative algorithms for Feature Selection

Data	CI-CDO	CDO	MGO	HBA	FHO	FPA	DNLPSO	GGSA	GWO_CS
D1	98.33	97.67	**98.36**	98.26	98.21	98.29	98.17	97.6	98.26
D2	75.53	74.13	75.45	75.03	75.17	74.9	74.88	74.1	**75.57**
D3	**100**	90.3	**100**	99.98	99.2	97.73	**100**	89.67	98.78
D4	87.39	84.49	**87.5**	86.02	86.76	87.16	86.08	83.75	86.48
D5	76.77	71.22	**76.85**	76.3	75.93	74.96	75.43	70.85	76.36
D6	**94.77**	91.14	94.43	94.32	93.75	92.5	93.41	87.27	94.55
D7	**87.5**	85	**87.5**	87.29	87.29	**87.5**	87.08	82.08	86.04
D8	77.12	72.28	77.2	**77.28**	75.77	76.17	75.98	72.15	77.15
D9	83.54	82.92	83.57	83.5	83.2	82.19	82.49	82.6	**83.61**
D10	**100**	99.41	**100**	99.95	**100**	99.73	99.64	95.91	**100**
D11	**100**	97.34	99.91	99.95	99.72	99.53	99.58	97.34	99.86
D12	**97.14**	94.53	96.53	96.82	95.76	94.66	94.96	93.07	96.87
D13	**89.15**	87.52	88.75	88.93	88.68	88.31	88.78	87.6	89.07
D14	**80.58**	72.95	78.69	80.19	78.88	75.8	77.98	73.65	80.26
D15	**90**	78.5	**90**	88.5	87.5	**90**	89.5	87	85
D16	**92.26**	82.34	89.68	90.24	88.87	88.87	91.21	86.69	87.02
D17	**64.92**	59.15	63.08	64.01	63.9	62.75	64.07	62.69	64.18
D18	**94.29**	88.1	92.86	93.33	92.14	92.38	93.57	90.71	90.71
D19	**89.61**	84.21	89.47	88.03	87.63	89.47	88.55	84.08	85.39
D20	**92.27**	77.73	88.18	90.23	88.64	85.45	90	80.23	81.14
D21	92.88	91.38	92.37	**93.15**	92.47	92.35	92.99	91.77	92.7

Table 14 Friedman Test Results for Feature Selection

Data	CI-CDO	CDO	MGO	HBA	FHO	FPA	DNLPSO	GGSA	GWO_CS
D1	3.63	7.93	**3.08**	4.33	4.65	3.45	4.38	8.28	5.30
D2	**2.83**	8.23	3.20	4.93	4.13	5.15	5.13	8.13	3.30
D3	3.23	8.00	**3.13**	3.43	5.23	5.40	**3.13**	8.85	4.63
D4	2.43	7.75	**2.18**	5.63	4.43	3.33	5.28	8.35	5.65
D5	2.78	8.60	**2.55**	3.60	4.40	6.15	4.90	8.35	3.68
D6	**2.85**	7.63	2.90	3.33	4.45	6.08	4.28	8.90	4.60
D7	**3.10**	7.90	**3.10**	3.35	4.40	3.70	3.78	8.50	7.18
D8	3.15	8.20	2.95	**2.93**	5.85	4.98	5.15	8.30	3.50
D9	2.85	6.43	3.05	2.95	4.93	8.03	6.05	8.13	**2.60**
D10	**2.60**	7.95	3.38	3.08	4.70	5.80	3.53	8.28	5.70
D11	**2.40**	8.65	2.95	2.58	5.03	6.03	4.03	7.75	5.60
D12	**2.03**	7.60	3.38	2.60	5.05	7.15	5.70	8.55	2.95
D13	**2.35**	8.75	3.88	3.73	4.50	6.40	3.88	7.63	3.90
D14	**2.15**	8.50	4.58	2.85	4.60	7.00	4.80	7.55	2.98
D15	**2.40**	8.60	**2.40**	3.75	4.75	5.33	3.63	6.38	7.78
D16	**2.05**	8.90	3.53	3.75	4.98	4.95	2.90	6.95	7.00
D17	**2.65**	9.00	5.53	3.38	3.70	6.50	3.50	6.65	4.10
D18	2.58	8.30	3.78	**2.50**	4.03	5.30	2.98	7.55	8.00
D19	2.58	7.90	**2.45**	3.68	4.15	5.25	4.15	7.65	7.20
D20	**2.50**	8.30	3.80	2.73	3.05	5.58	3.48	7.95	7.63
D21	3.00	8.55	5.30	**1.95**	4.55	5.75	2.60	8.25	5.05
Sum	**56.10**	171.65	71.05	71.00	95.53	117.28	87.20	166.90	108.30
Avg.	**2.67**	8.17	3.38	3.38	4.55	5.58	4.15	7.95	5.16

Table 15 Forecasting Performance of the Proposed CI-CDO-ANN Model on Egypt Climate Dataset

Metric	**Full**	**Short-Term**	**Long-Term**
Normalized RMSE	0.2183	—	—
RMSE (°C)	7.4211	7.5722	7.1219
MAE (°C)	5.8775	—	—
MAPE (%)	8.10	8.11	8.01
R^2 (Coefficient of Determination)	0.6579	—	—
NSE (Nash–Sutcliffe Efficiency)	0.6579	0.6333	0.6708
Correlation Coefficient (r)	0.8112	—	—

Table 16 Normalized RMSE values of the proposed CI-CDO compared with the CDO algorithm over 20 independent runs.

	CI-CDO		CDO	
fitness over 20 runs	0.21998561	0.21903833	0.22356233	0.22423621
	0.21815358	0.21876092	0.22353208	0.22353674
	0.21931450	0.22006833	0.22660040	0.22350747
	0.21825776	0.21818670	0.22977123	0.22379305
	0.21883338	0.22033625	0.22439690	0.22376680
	0.22010643	0.21889045	0.22363199	0.23406695
	0.21923333	0.21916261	0.23540623	0.22368913
	0.21703254	0.21933670	0.22454582	0.23025545
	0.21828514	0.22005533	0.22349138	0.22352995
	0.21843017	0.21756620	0.22360161	0.23961657
min	**0.21703254**		0.22349138	
Avg.	**0.21895171**		0.22642691	
max	**0.22033625**		0.23961657	

References

[1] Li, G., Chen, Z., Zhou, Y., Huang, Z., Xia, X.: Towards large-scale multi-objective feature selection: A two-stage evolutionary algorithm guided by dual feature weightings. Expert Systems with Applications **298**, 129823 (2026)

[2] Elmi, Z., Elmi, S., Danishvar, S.: Nrbo-agp: A novel feature selection approach for accurate protein solubility prediction. Expert Systems with Applications, 129194 (2025)

[3] Xia, M., Zhang, L., Li, K., Cheng, F.: Ods-ea: An objective to decision space-based evolutionary algorithm for high-dimensional feature selection. Expert Systems with Applications **303**, 130560 (2026)

[4] Chen, K., Liu, K., Wang, P., Liang, J., Yang, Z., Yu, K.: An efficient evolutionary feature selection algorithm with divide-and-conquer strategy for classification. Expert Systems with Applications **302**, 130416 (2026)

[5] Xue, B., Zhang, M., Browne, W.N., Yao, X.: A survey on evolutionary computation approaches to feature selection. IEEE Transactions on Evolutionary Computation **20**(4), 606–626 (2016)

[6] Ismail, F.A., Houssein, E.H., *et al.*: Chaos-enhanced equilibrium optimizer for feature selection in biomedical data. Expert Systems with Applications **195**, 116611 (2022)

[7] Ömer Faruk, D.: A hybrid neural network and arima model for water quality time series prediction. Engineering Applications of Artificial Intelligence **23**(4), 586–594 (2010)

[8] Houssein, E.H., Helmy, B.E., *et al.*: Metaheuristic optimization algorithms for machine learning: A comprehensive review. Discover Computing **3**(1), 1–22 (2024)

[9] Atef, A., El-Bassiouny, A., Houssein, E.H.: Metaheuristic-based hybrid models for predictive analytics: Applications in healthcare and beyond. Discover Computing **4**(1), 1–19 (2025)

[10] Lin, B., Miao, D., Liu, C., Zhang, H., Wang, R., Pedrycz, W.: Two-stage feature selection utilizing three-way adaptive neighborhood characteristic measure and optimal combination search. Expert Systems with Applications, 129650 (2025)

[11] Houssein, E.H.H., Saeed, M.K., Hu, G., al-sayed, M.: Metaheuristics for solving global and engineering optimization problems: Review, applications, open issues and challenges. Archives of Computational Methods in Engineering (2024)

[12] Zhou, L., Wang, W., Jin, G.: A multi-strategy enhanced reptile search algorithm for global optimization and engineering optimization design problems. Cluster

Computing (2024) https://doi.org/10.1007/s10586-024-04819-3 . Springer

[13] Wu, L., Li, S., Wu, F., *et al.*: Multi-strategy enhanced hiking optimization algorithm for task scheduling in the cloud environment. Journal of Bionic Engineering (2025) https://doi.org/10.1007/s42235-025-00674-z

[14] Yu, M., Xu, J., Liang, W., Qiu, Y., Bao, S., Tang, L.: Improved multi-strategy adaptive grey wolf optimization for practical engineering applications and high-dimensional problem solving. Artificial Intelligence Review **57** (2024) https://doi.org/10.1007/s10462-024-10821-3

[15] Ye, Z., Zeng, Y., Cai, T., Shen, J., Zhou, W., He, Q., Mei, M., Cen, Y.: Kmhbo: A knowledge-guided multi-niche hybrid breeding optimization algorithm for high-dimensional multimodal feature selection. Expert Systems with Applications **298**, 129911 (2026)

[16] Shi, Y., Li, P., Yuan, X., Wu, Y., Wang, H.: Multi-source multi-label feature selection with missing features. Expert Systems with Applications **298**, 129879 (2026)

[17] Santos, J.G.F., Silva, L.V., Marins, F.A.S.: Metaheuristics for time-series forecasting: a systematic literature review. International Journal of Forecasting **37**(2), 781–797 (2021)

[18] Alsmadi, M.K., Alsmirat, M.A., Rawashdeh, S.: A hybrid deep learning model for forecasting energy consumption in smart cities. Sustainable Cities and Society **78**, 103500 (2022)

[19] Wang, L., Lin, Z., Wong, A.: Deep learning for forecasting disease progression in patients with covid-19. IEEE Transactions on Neural Networks and Learning Systems **32**(8), 3462–3471 (2020)

[20] Holland, J.H.: Genetic Algorithms vol. 267, pp. 66–73. JSTOR, ??? (1992)

[21] Ibrahim, A.M., Tawhid, M.A.: In: Dey, N. (ed.) IGA: An Improved Genetic Algorithm for Real-Optimization Problem, pp. 105–138. Springer, Singapore (2023)

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

[23] Ibrahim, A.M., Tawhid, M.A.: A hybridization of cuckoo search and particle swarm optimization for solving nonlinear systems. Evol. Intel. **12**(4), 541–561 (2019)

[24] Storn, R., Price, K.: Differential evolution–a simple and efficient heuristic for

global optimization over continuous spaces. Journal of Global Optimization **11**(4), 341–359 (1997)

[25] Tawhid, M.A., Ibrahim, A.M.: A hybridization of grey wolf optimizer and differential evolution for solving nonlinear systems. Evolving Systems **11**, 65–87 (2020)

[26] Dorigo, M., Gambardella, L.M.: Ant colony optimization for continuous domains. European Journal of Operational Research **102**(1), 1–13 (1999)

[27] Karaboga, D.: A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. Journal of Global Optimization **39**, 459–471 (2007)

[28] Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S.: GSA: A gravitational search algorithm. Information Sciences **179**(13), 2232–2248 (2009)

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

[30] Tawhid, M.A., Ibrahim, A.M.: A hybridization of grey wolf optimizer and differential evolution for solving nonlinear systems. Evolving Systems **11**, 65–87 (2020)

[31] Mirjalili, S., Lewis, A.: The whale optimization algorithm. Advances in Engineering Software **95**, 51–67 (2016)

[32] Tawhid, M.A., Ibrahim, A.M.: Feature selection based on rough set approach, wrapper approach, and binary whale optimization algorithm. Int. J. Mach. Learn. & Cyber. **11**, 573–602 (2020)

[33] Heidari, A.A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., Chen, H.: Harris hawks optimization: Algorithm and applications. Future Generation Computer Systems **97**, 849–872 (2019)

[34] Faramarzi, A., Heidarinejad, M., Stephens, B., Mirjalili, S.: Equilibrium optimizer: A novel optimization algorithm. Knowledge-Based Systems **191**, 105190 (2020)

[35] Ali, M.Z., Abdullah, A., Zaki, A.M., Rizk, F.H., Eid, M.M., El-Kenawy, E.M.: Advances and challenges in feature selection methods: A comprehensive review. Journal of Artificial Intelligence and Metaheuristics **7**(1), 67–77 (2024)

[36] Wang, G.-G., Guo, L., Duan, H., Wang, H.: A chaotic particle swarm optimization algorithm for global numerical optimization. Soft Computing **23**(9), 3105–3122 (2019)

[37] Alatas, B.: Chaotic bee colony algorithms for global numerical optimization.

Expert Systems with Applications **37**(8), 5682–5687 (2010)

[38] Tanyildizi, E., Demir, G.: Chaotic whale optimization algorithm on large scale global optimization. Neural Computing and Applications **28**, 3017–3037 (2017)

[39] Qu, C., Liang, Y., Li, J.: Chaotic grey wolf optimizer for high-dimensional global optimization problems. Applied Soft Computing **102**, 107083 (2021)

[40] Ahmad, S., Ullah, A., Rashid, T.: Chaotic reptile search algorithm for feature selection and engineering design optimization. Engineering Applications of Artificial Intelligence **123**, 106179 (2023)

[41] Abdelrazek, M., Abd Elaziz, M., El-Baz, A.: CDMO: Chaotic dwarf mongoose optimization algorithm for feature selection. Scientific Reports **14**(1), 701 (2023)

[42] Alibabaei Shahraki, M.: Cloud drift optimization (CDO) algorithm: A nature-inspired metaheuristic. Discover Computing **4**(1), 1–19 (2024)

[43] Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of ICNN'95 - International Conference on Neural Networks, pp. 1942–1948 (1995). IEEE

[44] Liang, X., Sun, Y., He, G.: Chaotic particle swarm optimization for neural network training. Information Sciences **181**(9), 1686–1701 (2011)

[45] Mirjalili, S., Lewis, A.: Particle swarm optimization with chaotic sequences for global optimization. Applied Soft Computing **34**, 475–487 (2015)

[46] Chen, X., Li, Y., Zhao, J.: Chaotic grey wolf optimizer for engineering design. Computers & Structures **195**, 1–12 (2018)

[47] Singh, A., Sharma, P.: An improved chaotic grey wolf optimizer for multi-objective problems. Applied Soft Computing **75**, 313–328 (2019)

[48] Wang, X., Liu, Y., Zhang, Z.: Chaotic whale optimization algorithm for parameter optimization. Expert Systems with Applications **115**, 130–144 (2019)

[49] Faramarzi, A., Heidarinejad, M., Mirjalili, S.: Marine predators algorithm: A nature-inspired metaheuristic. Expert Systems with Applications **152**, 113377 (2020)

[50] Das, S., Kumar, P., Raj, A.: Chaotic marine predators algorithm for global optimization. Information Sciences **540**, 1–24 (2020)

[51] Ziarani, M., Vahdat, A.: Sand cat swarm optimization. Applied Soft Computing **89**, 106099 (2020)

[52] Singh, B., Prasad, S.: Chaotic sand cat swarm optimization. Swarm and Evolutionary Computation **58**, 100709 (2020)

[53] Li, Y., Wang, L., Zhang, J.: Heap-based optimizer for global optimization. Engineering Applications of Artificial Intelligence **97**, 104054 (2021)

[54] Wang, C., Zhang, J.: Chaotic search enhancement in heap-based optimizer. Applied Intelligence (2022)

[55] Yang, X.-S., Deb, S.: Cuckoo search via lévy flights. In: 2009 World Congress on Nature & Biologically Inspired Computing, pp. 210–214 (2009). IEEE

[56] Wang, L., Zhang, M., Li, H.: Chaotic cuckoo search algorithm for feature selection. Expert Systems with Applications **112**, 1–18 (2018)

[57] Abedinpourshotorban, H., Saberi, M., Ghanbarzadeh, A., Nadjar Araabi, B.: Electromagnetic field optimization: A physics-inspired metaheuristic optimization algorithm. Engineering Applications of Artificial Intelligence **87**, 103249 (2020) https://doi.org/10.1016/j.engappai.2019.103249

[58] Cao, F., Chen, J., Li, L.: Chaotic particle swarm optimization for feature selection. Neurocomputing **275**, 768–781 (2018)

[59] Singh, R., Singh, P.: Feature selection using chaotic pso. Information Sciences **521**, 26–47 (2020)

[60] Kumar, S., Singh, R., Sharma, A.: Chaotic ant lion optimizer for feature selection. Expert Systems with Applications **126**, 198–214 (2019)

[61] Singh, A.K., Kumar, M., Singh, P.: Chaotic selfish herd optimizer for feature selection. Applied Soft Computing **95**, 106435 (2020)

[62] Zhao, J., Wang, L., Chen, H.: Asymmetric chaotic maps in competitive swarm optimizer. Knowledge-Based Systems **197**, 105849 (2020)

[63] Mafarja, H., Mirjalili, B.: Chaotic butterfly optimization algorithm for feature selection. Applied Intelligence **49**(6), 2210–2230 (2019)

[64] Li, L., Huang, J., Zhang, X.: Chaotic firefly algorithm for feature selection. Neural Computing and Applications **31**, 6497–6515 (2019)

[65] Aljarah, Y., Faris, M., Abdullah, R.: Chaotic grey wolf optimizer for feature selection. Expert Systems with Applications **117**, 141–153 (2019)

[66] Zhang, J., Wang, Y., Sun, X.: Chaotic pso for time series forecasting. Neurocomputing **121**, 325–335 (2013)

[67] Yilmaz, M., Kaya, A.: Chaotic pso-based ann training for load forecasting. Energy

Conversion and Management **101**, 278–289 (2015)

[68] Pham, L.H., Nguyen, T.T.: Chaotic grey wolf optimizer in short-term load forecasting. Energy **239**, 122201 (2022)

[69] Chen, X., Zhao, Y., Li, J.: Rainfall prediction using chaotic whale optimization algorithm. Water **12**(5), 1234 (2020)

[70] Das, S., Kumar, P.: Wind speed forecasting using chaotic marine predators algorithm. Renewable Energy **171**, 973–987 (2021)

[71] Singh, B., Prasad, S.: Temperature forecasting using chaotic sand cat swarm optimization. Environmental Modelling & Software **125**, 104603 (2020)

[72] Azizi, M., Aickelin, U., Khorshidi, H., Baghalzadeh, M.: Energy valley optimizer: a novel metaheuristic algorithm for global and engineering optimization. Scientific Reports **13**(1), 226 (2023) https://doi.org/10.1038/s41598-022-27344-y

[73] Li, Y., Zhang, X., Chen, B.: Crane search algorithm with chaotic initialization. Soft Computing **28**(3), 1221–1235 (2024) https://doi.org/10.1007/s00500-024-08800-1

[74] Wu, T., Han, J., Liu, Y.: A chaotic-enhanced squirrel search algorithm for engineering design problems. Applied Soft Computing **140**, 111040 (2025)

[75] Abualigah, L., Yousri, D.S.A., Abd Elaziz, M., Ewees, A.A., Al-Qaness, M.A.A., Ghoneim, A., Snasel, V.: The mountain gazelle optimizer: A novel nature-inspired metaheuristic algorithm. Expert Systems with Applications **184**, 115402 (2021)

[76] Hashim, F.A., Houssein, E.H., Mabrouk, M.S., Al-Atabany, W., Mirjalili, S.: Honey badger algorithm: New metaheuristic algorithm for solving optimization problems. Mathematics and Computers in Simulation **192**, 84–110 (2022)

[77] Yousri, D.S.A., Abualigah, L., Abd Elaziz, M., Al-Qaness, M.A.A., Ewees, A.A., Ghoneim, A., Snasel, V.: Fire hawk optimizer: A novel nature-inspired metaheuristic optimization algorithm. Engineering Applications of Artificial Intelligence **114**, 105082 (2022)

[78] Yang, X.-S.: Flower pollination algorithm for global optimization. In: Unconventional Computation and Natural Computation (UCNC), pp. 240–249. Springer, ??? (2012)

[79] Cheng, S., Wang, J., Liu, Y., Li, X., Wang, W.: A dynamic neighborhood learning based particle swarm optimization algorithm. IEEE Access **10**, 6414–6430 (2022)

[80] Aljarah, I., Faris, H., Mirjalili, S.: An improved gbest-guided gravitational search algorithm based on adaptive strategies for feature selection. Information Sciences **481**, 175–190 (2019) https://doi.org/10.1016/j.ins.2018.12.006

[81] Xu, H., Liu, X., Su, J.: An improved grey wolf optimizer algorithm integrated with cuckoo search. In: Proceedings of the 2017 36th Chinese Control Conference (CCC), pp. 10543–10548 (2017). IEEE

[82] Mirjalili, S., Mirjalili, S.M., Lewis, A.: Grey wolf optimizer. Advances in Engineering Software **69**, 46–61 (2014) https://doi.org/10.1016/j.advengsoft.2013.12.007

[83] Yang, X.-S., Deb, S.: Cuckoo search: Recent advances and applications. Neural Computing and Applications **24**(1), 169–174 (2013) https://doi.org/10.1007/s00521-013-1367-1

[84] Suganthan, P.N., Mohamed, A.W., Hadi, A.A., Mohamed, A.K., Agrawal, P., Kumar, A.: Problem Definitions and Evaluation Criteria for the CEC 2021 Special Session and Competition on Single Objective Bound Constrained Numerical Optimization (2020)

[85] Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics Bulletin **1**(6), 80–83 (1945)

[86] Gibbons, J.D., Chakraborti, S.: Nonparametric Statistical Inference, 5th edn. Chapman & Hall/CRC, ??? (2011)

[87] Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research **7**, 1–30 (2006)

[88] Dua, D., Graff, C.: UCI Machine Learning Repository (2019). http://archive.ics.uci.edu/ml

[89] Atef, D., El-Shahat, D.: Metaheuristic techniques in feature selection: A concise review. International Journal of Computers and Informatics (Zagazig University) **7**, 1–12 (2025)

[90] NOAA National Centers for Environmental Information: Global Historical Climatology Network - Daily (GHCN-Daily). https://www.ncei.noaa.gov/products/land-based-station/global-historical-climatology-network-daily. Accessed: 2025-09-02 (2025)

[91] KathituCodes: Climate-Change-in-Africa Dataset. https://github.com/KathituCodes/Climate-Change-in-Africa. Accessed: 2025-09-02 (2023)

[92] Shankar, S., Baskar, S.: Data normalization in neural networks: A comparative study. International Journal of Computer Applications **113**(9) (2015)

[93] Gandomi, A.H., Alavi, A.H., Yang, X.-S., Talatahari, S.: Chaos-enhanced accelerated particle swarm optimization. Communications in Nonlinear Science and Numerical Simulation **18**(2), 327–340 (2013)

[94] Li, X., Yang, X.-S., Zhang, Q.: Comparative analysis of swarm optimization algorithms in renewable energy forecasting. Renewable Energy **145**, 2560–2572 (2020)

[95] Aljarah, I., Faris, H., Mirjalili, S.: Evolutionary feature selection using harmony search algorithm for biomedical data classification. Information Sciences **425**, 93–109 (2018)

[96] Elshamy, M.E., Seierstad, I.A., Sorteberg, A.: Impacts of climate change on nile flows at dongola using statistical downscaling. Hydrology and Earth System Sciences **13**(5), 693–714 (2009)