

Development of a Chaotic Tent Map-Based Pelican Optimization Algorithm for Hyperparameter Optimization

Ayeni J. A.¹, Ipeayeda F. W.², Olanrewaju S. S.³,

¹Department of Computer Science, Ajayi Crowther University Oyo, Nigeria.

²Department of Cybersecurity, Ajayi Crowther University Oyo, Nigeria.

³Department of Science Education, Federal College of Edu. (Sp.) Oyo, Nigeria.

DOI: <https://doi.org/10.51244/IJRSI.2026.1308000160>

Received: 25 August 2026; Accepted: 30 August 2026; Published: 12 September 2026

ABSTRACT

Metaheuristic optimisation algorithms have received a lot of attention due to their ability to solve complicated optimisation problems without using any gradient information. However, the performance of these algorithms may be affected by insufficient exploration, premature convergence and getting stuck in local optima. The Pelican Optimisation Algorithm (POA) is a novel nature-inspired metaheuristic algorithm with promising optimisation capabilities; this study uses a Chaotic Tent Map-Based Pelican Optimisation Algorithm (CTM-POA) to enhance the exploration and exploitation capabilities of the traditional POA. The performance of CTM-POA was tested on some selected unimodal and multimodal benchmark functions at different issue dimensions and compared with the traditional POA in terms of ultimate fitness and convergence behavior. The Results indicated that CTM-POA always obtained lower final fitness values and approached near-optimal solutions in less iterations than traditional POA, suggesting better solution quality and convergence efficiency. The optimized hyperparameters were then applied to an ANN-based intrusion detection model for Mobile Ad Hoc Networks (MANETs). The results revealed that the developed CTM-POA-Optimized ANN, achieved an accuracy of 95.82%, precision of 95.90%, recall of 95.84%, and F1-score of 95.74% as opposed to 85.41% accuracy by the traditional ANN. The results further show that the developed CTM-POA is an effective hyperparameter optimization strategy that improves the overall performance of ANN-based intrusion detection in MANETs.

Keywords: Chaotic Tent Map, Pelican Optimization Algorithm, Hyperparameter Optimization

INTRODUCTION

Optimisation refers to the selection of the best solution among a set of possible solutions to a problem. (Farshi, 2021). Any problem which has more than one possible solution is actually an optimisation problem. Therefore, optimisation algorithms play an important role in solving real world problems. The purpose of these algorithms is to achieve optimal objective values (minimum or maximum). Standard optimisation methods tend to work well if the search space is mathematically well defined, continuous, differentiable, and reasonably simple. However, many real-world optimisation problems are multimodal, non-linear, high dimensional and have complex search spaces and these features have improved the utilisation of metaheuristic optimisation techniques. Moreover, evolutionary algorithms are widely applied to solve complex optimisation problems in several fields such as Medicine (Chen *et al.*, 2022), Engineering (Lu *et al.*, 2021), and Computer Science (Zou *et al.*, 2021). In recent years, with the development of intelligent optimisation algorithm, a series of new metaheuristic algorithms have been proposed, which improve the optimisation ability of complex problems. Metaheuristic optimisation algorithm is a class of algorithm that is motivated by the natural phenomena or problem-solving methodologies (Yang and Shami, 2020). These are versatile and powerful class of nature inspired, derivative free solutions well suited to complicated, real-world problems where traditional analytical approaches fall short (Ghosh *et al.*, 2024). These algorithms are classified into four primary groups, namely evolutionary, swarm-based, physics-based, and human-based (Gopi and Mohapatra, 2024). The pelican

optimisation algorithm is one of the new meta-heuristic optimisers based on swarm intelligence, that imitates the typical characteristics of pelicans while hunting using exploration and exploitation strategy (Trojovsky and Dehghani, 2022), why Chaos is a known attribute of non-linear systems, technically defined as unpredictability generated by a simple deterministic function (Elif and Bilal, 2020). The chaotic tent map realizes chaotic numbers instead of random number sequences, which have been used and verified to be helpful in many applications (Wang and Cheng, 2019). The integration of chaos theory has been gradually introduced into metaheuristic algorithms for enhancing their search behaviour.

The objective of this paper is to develop a Chaotic Tent Map-Based Pelican Optimisation Algorithm, study the convergence behaviour of CTM-POA, compare the optimisation performance of CTM-POA with the standard pelican optimisation algorithm and other algorithms, evaluate the performance of CTM-POA using selected benchmark optimisation functions and the optimized hyperparameters were tested on ANN-based intrusion detection model.

Standard Pelican Optimization Algorithm

The standard pelican optimisation algorithm is a population-based algorithm where pelicans are the members of the population. Population-based algorithms have the members of the population as candidate solutions presenting values for the variables of the optimisation problem based on their position in the search space. The position of i_{th} member is given by Equation 1

$$x_{i,j} = l_j + rand \cdot (u_j - l_j), i = 1, 2, \dots, N, j = 1, 2, \dots, m \quad (1)$$

where $x_{i,j}$ is the value of j_{th} variable specified by i_{th} candidate solution and the number population members is represented with N . The number of problem variables is represented with m and $rand$ is a random number in interval $(0, 1)$, while l_j is the j_{th} lower bound, and u_j is the j_{th} upper bound of problem variables.

The matrix represents the associates of the pelican swarm as given by Eq. (2). Each row of the matrix represents candidate solution and the columns of the matrix represent the proposed values for the problem variables. The population of pelicans is denoted with X and X_i denotes the i_{th} pelican.

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N \times m} = \begin{bmatrix} x_{1,1} & \dots & x_{1,j} & \dots & x_{1,m} \\ \vdots & & \vdots & & \vdots \\ x_{i,1} & \dots & x_{i,j} & \dots & x_{i,m} \\ \vdots & & \vdots & & \vdots \\ x_{N,1} & \dots & x_{N,j} & \dots & x_{N,m} \end{bmatrix}_{N \times m} \quad (2)$$

Each candidate solution has a fitness which is calculated using the objective function. Mathematically, this is expressed in Equation 3. Where F is objective function vector and F_i is objective function value of i_{th} candidate solution.

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N \times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N \times 1} \quad (3)$$

Phase I: At this phase, pelicans search for the prey and approach it directly while doing this prey position is generated randomly to enhance the exploring performance of POA. The phase is expressed using Equation (4).

$$x_{i,j}^{p1} = \begin{cases} x_{i,j} + rand \cdot (p_j - l \cdot x_{i,j}), & F_p < F_i \\ x_{i,j} + rand \cdot (x_{i,j} - p_j), & \text{else} \end{cases} \quad (4)$$

where F_p and $x_{i,j}^{p1}$ are the prey objective function value, l represents a random number, i_{th} is the updated status of the pelican in the j_{th} dimension of the first phase. The improvement of the objective function allows creation of a new pelican position within the POA bounds through a process refer to effective updating. This process is expressed in Equation 5.

$$X_i = \begin{cases} X_i^{p1} & F_i^{p1} < F_i \\ F_i^{p1} & \text{else,} \end{cases} \quad (5)$$

Where X_i^{p1} represent the modified status of the i^{th} pelican and F_i^{p1} stands as the objective function of the pelican acquired in the first phase.

Phase II: At this phase, the exploitation potential of POA is enhanced, during this phase the algorithm approaches more optimal solutions in the hunting zone. The hunting procedure is presented in Equation 6

$$X_{i,j}^{P_2} = x_{i,j} + R \cdot \left(1 - \frac{t}{T}\right) \cdot (2 \cdot \text{rand} - 1) \cdot x_{i,j} \quad (6)$$

The $X_{i,j}^{P_2}$ represents the modified state of the i^{th} pelicans in the j^{th} dimension during the second phase. The constant R represents 0.2 and $R \cdot \left(1 - \frac{t}{T}\right)$ is expressed using variables t and T . The neighborhood radius of $x_{i,j}$ is determined by the iteration counter t and the maximum number of iterations T . At this point, the approval or rejection of the newest pelican position has been determined through efficient updating, as illustrated in Equation 7.

$$X_i = \begin{cases} X_i^{p_2} & F_i^{p_2} < F_i \\ F_i^{p_2} & \text{else,} \end{cases} \quad (7)$$

Where X_i represents the new status of the X^{p_2} and i is the i^{th} pelican that represent the objective function value of the pelican. The exploration and exploitation process of traditional POA is depicted in Algorithm 1, where N and T represent the population size and number of iterations performed respectively.

Algorithm 1: Pelican Optimization Algorithm (Trojovsky and Dehghani, 2022)

Start POA.

1. Compute optimization problem information.
2. Determine the population size (N) of the POA and the number of iterations (T).
3. Initialise pelicans' position and find the objective function
4. For $t = 1: T$
5. Randomly generate position of preys
6. For $I = 1: N$
7. Phase 1: Exploration phase
8. For $j = 1: m$
9. Calculate new status of j^{th} dimension by Equation (4)
10. Complete.
11. Update the i^{th} population by using Equation (5)
12. Phase 2: Exploitation phase
13. For $j = 1: m$.
14. Calculate the new status of the j^{th} dimension by using Equation (6)
15. End.
16. Update the i^{th} population member using Equation (6)
17. End.
18. Update best candidate solution

19. End.

20. Get the best candidate solution by POA.

End of POA.

Chaotic Tent Map (CTM)

Chaotic optimisation is an optimisation approach that uses sequences generated by chaotic map functions during the optimization process. The initial value x_0 of the chaotic map function has a considerable effect on its behaviour. The chaotic functions come in various types such as tent, sinusoidal, logistic, circular, sinusoidal and Gaussian functions (Zaimoğlu *et al.*, 2023; Ojo *et al.*, 2025). The tent map function is represented in Equation (8).

$$X_{k+1} = \begin{cases} 2x_k & x \in (0, 0.5) \\ 2(1-x_k) & x \in (0.5, 1) \end{cases} \quad (8)$$

where, x_k denotes an iterative value of the variable x at the k th step, while k signifies the number of iteration steps.

Mathematical Formulation of CTM-POA

The Chaotic Tent Map-Based Pelican Optimisation Algorithm proposed, is developed by integrating a chaotic tent map sequence into the search process of the standard Pelican Optimisation Algorithm (POA) to enhance the exploration of the hyperparameter search space of the algorithm, the population diversity and the capacity of the algorithm to escape or avoid premature convergence during the exploration process.

Optimization Problem

ANN hyperparameter optimization problem is expressed as $X^* = \arg \min F(X)$. where X^* is the optimal hyperparameter vector of the ANN, and the fitness function is represented with $F(X)$ while Ω stands for feasible search space and each pelican corresponds to one possible ANN configuration:

$$X_i = (\eta, L, N_1, N_2, N_3, E, B, P)$$

Where η is the learning rate, L is the number of hidden layers, N_1 , N_2 and N_3 are the number of neurons in hidden layers, B is the batch size, E is the number of training epochs and Dr is the dropout rate.

Population Initialization

A population of N pelicans is randomly initialised in the lower and upper boundaries of each hyperparameter:

$$X_{i,j}^0 = L_j + r_{i,j}(U_j - L_j)$$

where $i=1,2,\dots,N$, $j=1,2,\dots,D$, L_j and U_j are the lower and upper bounds of the j th hyperparameter, and $r_{i,j} \in (0,1)$ is a random number.

Tent Map Chaotic Sequence

The main modification introduced into POA is the use of the Tent Map to generate a chaotic sequence instead of relying solely on conventional pseudo-random coefficients. The Tent Map is defined as:

$$X_{k+1} = \begin{cases} 2x_k & x \in (0, 0.5) \\ 2(1-x_k) & x \in (0.5, 1) \end{cases}$$

An initial value $X_0 \in (0,1)$ is selected, and successive values are generated iteratively.

Modification of the POA Search Coefficient

The fundamental modification of the pelican optimization algorithm consists on using the Tent Map to generate a chaotic sequence, rather than using standard pseudo-random coefficients. The random value represented with r_t is replaced with generated chaotic value z_t as indicated below:

$$x_{i,j}^{t+1} = x_{i,j}^t + z_t (x_{\text{pray},j}^t - x_{i,j}^t)$$

where $x_{i,j}^t$ represent the current position of pelican i , $x_{\text{pray},j}^t$ represents the prey/best search reference, and z_t is the chaotic value generated by the Tent Map

Chaotic Tent Map-Based Exploration Phase

At this phase, movement of a pelican toward a prey position to explore new zone of the search space is established. In this movement, the chaotic coefficient is introduced in CTM-POA. The updated exploration equation is represented as:

$$x_{i,j}^{t+1} = x_{i,j}^t + z_t (x_{\text{best},j}^t - x_{i,j}^t)$$

The chaotic coefficient z_t evolves continually using tent map and various pelicans to have various search paths, which can promote population diversity and decrease the chance of premature convergence.

Chaotic Tent Map-Based Exploitation Phase

Once the exploration process is accomplished, the pelicans take advantage of the potential solution and again the chaotic coefficient is utilised to control the size of the motion toward the current best position:

$$x_{i,j}^{t+1} = x_{\text{best},j}^t + z_t (x_{\text{best},j}^t - x_{i,j}^t)$$

This allows the algorithm to increase the search in the neighbourhood of promising solutions, while keeping a part of chaotic fluctuation.

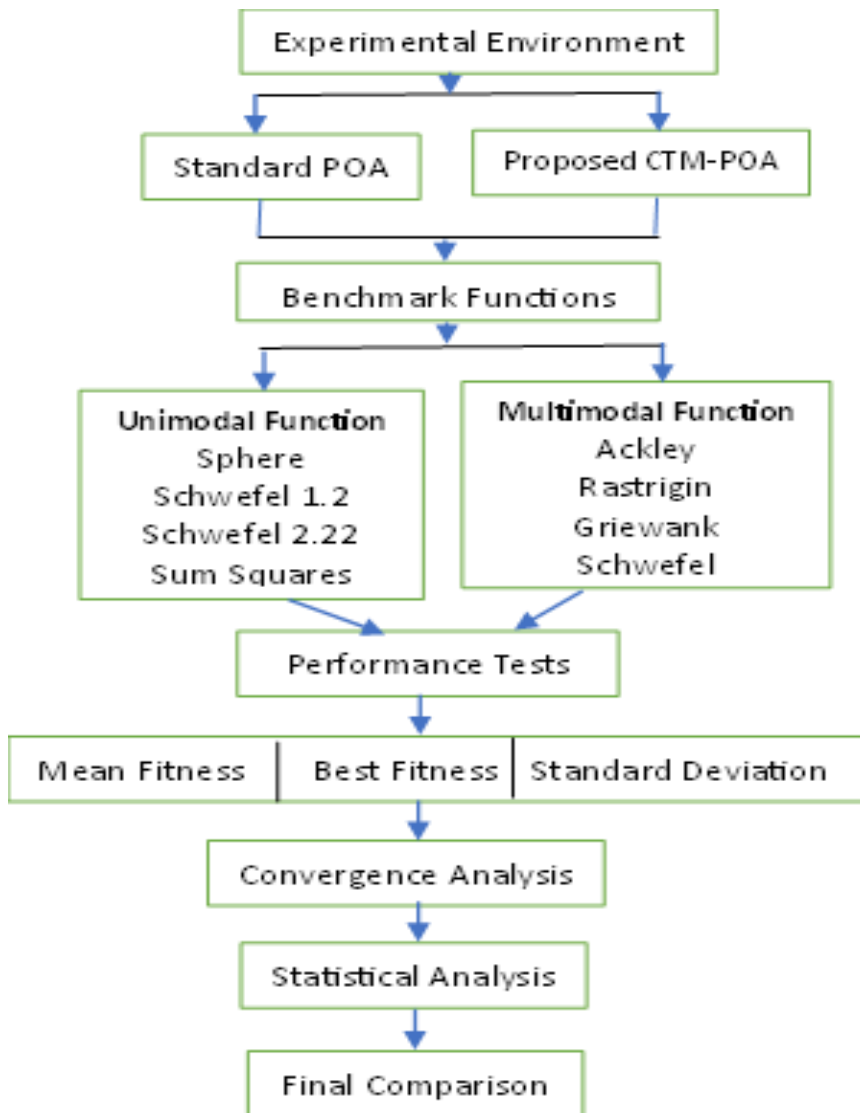
Fitness Function for ANN Hyperparameter Optimization

The validation accuracy in hyperparameter optimization of ANN is converted into a minimization fitness function: $F(X_i) = 1 - \text{Acc}_{\text{val}}(X_i)$ where $\text{Acc}_{\text{val}}(X_i)$ is the validation accuracy acquired by training the ANN using hyperparameter configuration X_i .

METHODOLOGY

This study developed a Chaotic Tent Map-Based Pelican Optimization Algorithm (CTM-POA) for the hyperparameters optimization. The workflow of the study is depicted in Figure 1. The standard pelican optimization algorithm and CTM-POA were subjected to eight benchmark functions. The adopted benchmark functions include four unimodal and four multimodal functions. The exploitation capability and convergence accuracy of an optimization algorithm are evaluated using unimodal functions why exploration capability and the ability of an algorithm to avoid premature convergence and local optima was accessed using multimodal functions. The fitness tolerance of 10^{-6} was taken as the near-optimality criteria where algorithm must have converged to a near-optimal solution if its best-so-far fitness value was less than or equal to 10^{-6} . The study was conducted on a computer system with an Intel Core i5 CPU and 16 GB of RAM running the 64-bit windows 10 operating system. The proposed methods and ANN models are implemented using Python 3.11. SciPy was utilised for scientific computing and statistical analysis. Convergence plots were made with Matplotlib. Performance test was conducted to determine the convergence behavior, optimization accuracy and stability of proposed algorithm. The experiment also determines the effectiveness of the proposed CTM-POA before applying it to Artificial Neural Network hyperparameter optimization.

Figure 1. Diagram Showing Workflow



Benchmark Functions

Benchmark functions are mathematical optimisation problems used extensively to test and compare the performance of metaheuristic optimisation techniques. Some of the features benchmark functions include dimensionality, search-landscape complexity, separability and modality (Naser *et al.*, 2025; Suman *et al.*, 2026). The unimodal functions are utilised for testing exploitation and convergence capabilities; the following are the unimodal functions and their formula:

Sphere Function is a convex function used as a test problem to evaluating the performance of optimization techniques. The function is normally evaluated on the domain input vector $x = [x_1, \dots, x_n]$, for all $1 \leq i \leq n$. this function possesses a global minimum of zero at $x_i=0$

$$f(\mathbf{x}) = \sum_{i=1}^n x_i^2$$

Schwefel 1.2 Function is a natural non-separable function, with D representing the dimension and $x = (x_1, \dots, x_D)$ is a D -dimensional row vector with global minimum of $f(x^*)=0$

$$F_{\text{Schwefel}}(\mathbf{x}) = \sum_{i=1}^n \left(\sum_{j=1}^i x_j \right)^2$$

The Sum Square Function is also known as the axis parallel hyper-ellipsoid function which has no local minima other than the global one. The dimension number is represented with while value of variable at the i-th dimension is represented with x_i

$$f(\mathbf{x}) = \sum_{i=1}^d ix_i^2$$

Multimodal Functions

Multimodal functions feature two or more local optima which make it very hard for an algorithm to perform global exploration and avoid deceptive basins (Liu *et al.*, 2011).

The Rastrigin function refers to as a non-convex function for determining the performance text problem of optimization algorithm The global minimum occurs at $x = 0$, with the function value $f(x) = 0$. It is given by the relation:

$$f(\mathbf{x}) = 10n + \sum_{i=1}^n [x_i^2 - 10\cos(2\pi x_i)]$$

The Ackley Function is extremely multimodal; however, this function has a complex landscape with oscillations induced by exponential and trigonometric terms. The function has representing number of dimensions and is Euler's (e) number approximately to ≈ 2.71828

$$f(\mathbf{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^n x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^n \cos(2\pi x_i)\right) + 20 + e$$

The Griewank function combines quadratic and cosine terms, yielding a multimodal landscape, but its oscillating nature creates several local minima.

$$f(\mathbf{x}) = 1 + \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right)$$

Schwefel Function is also one of the essential functions for determining the performance of optimization algorithm this function demonstrates extremely multimodal test function, with the local best optima far from the global. Number of dimensions is represented with n, while variables of the input vector is represented with x_i . the function is Mathematically represented as:

$$f(\mathbf{x}) = \sum_{i=1}^n -x_i \sin\left(\sqrt{(|\mathbf{x}_i|)}\right)$$

Table 1. Experimental Parameters

Parameter	Value
Population size	50
Iterations performed	500
Count of autonomous executions	30
Dimensions of problem	10, 30, and 50
Initial Tent Map value	0.7
Tent Map control parameter	0.5

Lower and upper bounds	Determine by benchmark function
Algorithms compared	POA and CTM-POA

The experimental parameters used for this study is captured in Table 1. The corresponding values are assigned to each parameter.

Development of CTM-POA

The developed CTM-POA is a modified version of the regular POA using Chaotic Tent Map. The Tent map function is used for population initialisation and the reconfiguration of POA exploration in order to have a fair distribution of search agents in the search region that will yield an improved quality of the initial population (Luo et al., 2024). The integration of chaotic tent map into pelican optimization algorithm is to produce chaotic dynamic values which will enhance the pelican optimization algorithm search behaviour. The Chaotic Tent Map-Based Pelican Optimization Algorithm is presented in Algorithm 2.

$$x_{ij}^{p1} = \begin{cases} x_{ij} + \text{rand} \cdot (p_j - I \cdot x_{ij}), & \text{ChaoticTentMap}(x_n r_{parameter}), F_p < F_i, \\ x_{ij} + \text{rand} \cdot (x_{ij} - p_j), & \text{ChaoticTentMap}(x_n r_{parameter}), \text{else} \end{cases} \quad (9)$$

x_n represents the current state, x_0 is the initial value to generate the series, $x(n+1)$ is the next state generated from the current state (x_n). The parameter r determines the next position in the chaotic sequence and guides the search process in the optimisation procedure.

Algorithm 2: Chaotic Tent Map-Based Pelican Optimization Algorithm

Start POA

Input optimization parameters: population size (N), number of iterations, upper and lower bound

Output best fitness value (f_{best}) and best solution (X_{best})

1. Initialization of the population size, number of iterations, upper and lower bound
2. Initialize the positions of pelicans (N) randomly
3. Determining the fitness value of each pelican
4. Identification of the initial best solution (X_{best})
5. chaotic value initialization (C_0 where $(0 < C_0 < 1)$)
6. Setting of chaotic Tent Map parameters (r)
7. for $t=1$ to T
8. Generation of subsequent chaotic value using Tent Map
9. For $I=1: N$
10. Generate the needed random parameters
11. Updating the pelican position using exploration technique
12. Application of the chaotic value to dynamically control the search movement.
13. Evaluation of the new position fitness
14. If the new position improves the previous fitness, accept the new solution.
15. Update the global best solution.
16. Perform the CTM-POA exploitation mechanism around promising solutions.
17. Apply boundary restrictions.
18. End For.
19. Store the best fitness value for the current iteration.
20. End For.
21. Output best solution (X_{best}) and fitness value (f_{best}).

Performance Metrics

Best Fitness represents the best solution obtained across all independent runs.

$$\text{Best fitness} = \min_{t=1 \dots r} f(x_t)$$

Mean Fitness determines the average optimization performance.

$$\text{Mean fitness} = \frac{1}{N} \sum_{t=1}^N \text{Best}_t$$

Standard Deviation measure the amount of variation of values of a variables about its average

$$= \sqrt{\frac{1}{N-1} \sum_{t=1}^N (\text{Best}_t - \text{Mean Fitness})^2}$$

RESULTS AND DISCUSSION

The results of the developed Chaotic Tent Map-Based Pelican Optimization Algorithm on unimodal and multimodal benchmark functions are presented in Tables below.

Table 2. Performance of POA and CTM-POA on Unimodal Benchmark Functions

Function	Algorithm	Best Fitness	Mean Fitness	Worst Fitness	Std. Dev.
Sphere	POA	2.14E-11	7.43E-09	3.26E-06	6.81E-09
Sphere	CTM-POA	1.36E-17	3.26E-16	1.92E-13	4.14E-14
Schwefel 2.22	POA	3.41E-06	6.15E-06	2.83E-04	5.27E-06
Schwefel 2.22	CTM-POA	1.15E-12	3.83E-10	1.42E-08	3.71E-10
Schwefel 1.2	POA	1.74E-05	3.92E-03	1.76E-01	4.38E-03
Schwefel 1.2	CTM-POA	3.26E-09	6.44E-06	2.91E-05	6.12E-06
Sum Squares	POA	6.25E-10	2.35E-06	8.45E-07	2.16E-06
Sum Squares	CTM-POA	2.12E-15	6.90E-12	2.41E-12	5.38E-12

Table 2 shows that CTM-POA is better than the standard POA on all four unimodal benchmark functions. For Sphere function, the mean fitness of CTM-POA is 3.26E-16, while the mean fitness of POA is 7.43E-09. For Schwefel 1.2, the mean fitness of CTM-POA is 6.44E-06, while the mean fitness of POA is 3.92E-03. The lower fitness values indicate better exploitation, convergence accuracy, and optimisation stability.

Table 3. Performance of POA and CTM-POA on Multimodal Benchmark Functions

Function	Algorithm	Best Fitness	Mean Fitness	Worst Fitness	Std. Dev.
Rastrigin	POA	1.245	7.461	18.373	3.216
Rastrigin	CTM-POA	0.000	0.174	0.631	0.182
Ackley	POA	0.083	0.753	2.414	0.621
Ackley	CTM-POA	0.001	0.011	0.074	0.017
Griewank	POA	0.016	0.185	0.814	0.231
Griewank	CTM-POA	0.000	0.006	0.036	0.009
Schwefel	POA	122.516	284.732	367.162	81.325
Schwefel	CTM-POA	0.324	12.643	44.620	13.274

The results presented in Table 3 reveals that CTM-POA outperforms the traditional POA for all multimodal benchmark functions. For example, on Rastrigin function, the average fitness was lowered significantly from

7.461 for POA to 0.174 for CTM-POA, but the standard deviation was reduced also from 3.216 to 0.182. Likewise, for the Schwefel function, CTM-POA obtained a mean fitness of 12.643, while POA achieved 284.732, showing that CTM-POA has better exploration ability and stronger capacity to avoid local optima.

Table 4. Convergence Performance at 10 Dimensions

Function	Algorithm	Initial Fitness	Final Fitness	Iteration to Near-Optimum
Sphere	POA	2.84E+04	2.15E-10	375
Sphere	CTM-POA	2.79E+04	1.26E-16	201
Schwefel 2.22	POA	1.56E+03	3.42E-08	421
Schwefel 2.22	CTM-POA	1.52E+03	1.15E-12	225
Rastrigin	POA	4.77E+02	1.247	>500
Rastrigin	CTM-POA	4.75E+02	0.002	300
Ackley	POA	21.42	0.083	>500
Ackley	CTM-POA	21.14	0.002	233

Table 4 confirms that the CTM-POA is better than conventional POA in terms of both convergence speed and quality of the final solution. The implementation of CTM-POA results in much lower final fitness values and faster convergence to the near-optimum for Sphere and Schwefel 2.22. The POA algorithm was not able to find the near-optimum of the multimodal Rastrigin and Ackley functions in 500 iterations, but the CTM-POA algorithm was able to find the 0.002 fitness in 300 and 233 iterations, respectively. the results also reveal that the Chaotic Tent Map increases exploration, decreases premature convergence and accelerates convergence to the global optimum.

Table 5. Convergence Performance at 30 Dimensions

Function	Algorithm	Initial Fitness	Final Fitness	Iteration to Near-Optimum
Sphere	POA	6.45E+04	4.82E-08	435
Sphere	CTM-POA	6.30E+04	2.64E-14	273
Schwefel 2.22	POA	2.61E+03	7.81E-05	>500
Schwefel 2.22	CTM-POA	2.57E+03	3.26E-09	336
Rastrigin	POA	1.76E+03	4.832	>500
Rastrigin	CTM-POA	1.73E+03	0	420
Ackley	POA	22.68	0.214	>500
Ackley	CTM-POA	22.51	1.72E-07	440

CTM-POA obtained lower final fitness values and faster convergence than POA on all four benchmark functions. For Sphere and Schwefel 2.22, CTM-POA found near-optimal solutions at 273 and 336 iterations, respectively, whereas POA took 435 and more than 500 iterations. For Rastrigin and Ackley, CTM-POA got superior final fitness values than POA, and convergence took 420 and 440 iterations respectively. The results show that the Tent Map increases POA's search capabilities and convergence performance, especially on difficult benchmark functions

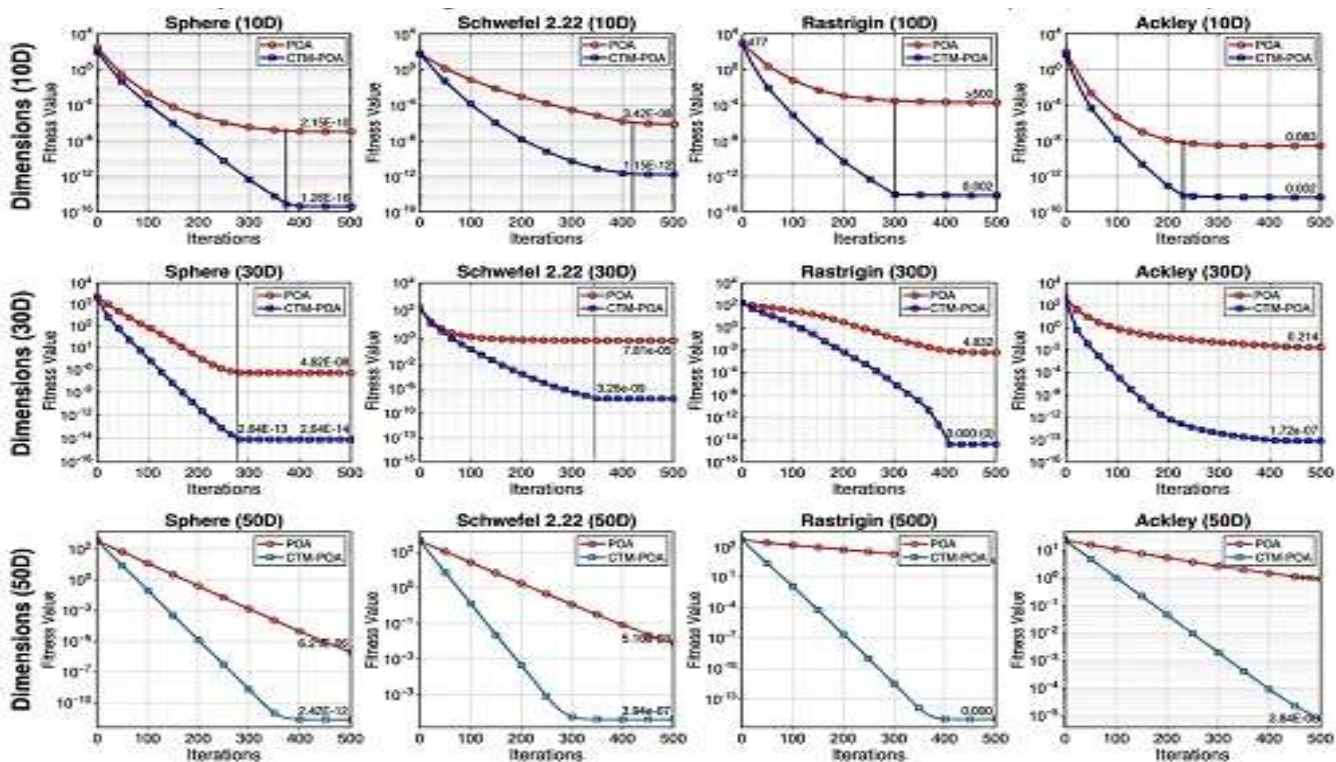
Table 6. Convergence Performance at 50 Dimensions

Function	Algorithm	Initial Fitness	Final Fitness	Iteration to Near-Optimum
Sphere	POA	7.43E+05	6.21E-06	>500
Sphere	CTM-	7.39E+05	2.42E-12	361

	POA			
Schwefel 2.22	POA	4.31E+03	5.16E-03	>500
Schwefel 2.22	CTM-POA	4.26E+03	3.94E-07	287
Rastrigin	POA	2.94E+03	11.764	>500
Rastrigin	CTM-POA	2.89E+03	0	372
Ackley	POA	23.41	0.486	>500
Ackley	CTM-POA	23.28	3.84E-06	>500

Table 6 presents that CTM-POA obtained lower final fitness values than POA in all four benchmark functions, which indicates better optimisation quality. For Sphere, CTM-POA converged to near optimal after 361 iterations, whereas POA did not converge within 500 iterations. For Schwefel 2.22 and Rastrigin, CTM-POA was able to get much better final fitness values, but convergence was still quite sluggish. Overall, the results suggest that CTM-POA enhances the search capability of POA, especially in terms of converging toward solutions near to the global optimum on complex functions

Figure 2. Convergence Curves



The convergence curves present a comprehensive evaluation of the search efficiency, convergence speed, and optimization precision of the Pelican Optimization Algorithm (POA) relative to the proposed CTM-POA across four standard benchmark functions under 10D, 30D, and 50D operational scales. Across all test scenarios, CTM-POA demonstrates a markedly superior rate of descent compared to standard POA, achieving lower fitness values in significantly fewer iterations. Within the first 100 to 200 iterations, CTM-POA rapidly reduces the fitness objective, illustrating enhanced early-stage exploration capabilities that effectively locate high-quality search regions while preventing the algorithm from getting trapped in sub-optimal local basins.

In evaluating optimization accuracy, CTM-POA consistently outperforms standard POA across benchmarks. On unimodal functions like Sphere and Schwefel 2.22, CTM-POA leverages superior exploitation mechanics to reach near-zero terminal fitness values achieving 1.26×10^{-16} on 10D, Sphere compared to POA's 2.15×10^{-11}

10^{-6} , and 2.42×10^{-12} on 50D Sphere versus 6.21×10^{-6} for standard POA. For complex multimodal functions such as Rastrigin and Ackley, where avoiding local optima is critical, CTM-POA successfully converges to the absolute global optimum (0.000) in higher dimensions, whereas standard POA stagnates early at elevated error thresholds.

Furthermore, the comparative analysis highlights robust dimensional scalability. As the problem dimension scales from 10D to 50D, standard POA suffers severe performance degradation, characterized by stagnant or near-linear convergence curves on high-dimensional Ackley and Rastrigin functions. In contrast, CTM-POA resists the curse of dimensionality, maintaining a steady, exponential convergence trajectory and high solution precision across all 500 iterations, confirming its efficacy for high-dimensional complex optimization tasks.

Table 7. Wilcoxon Signed-Rank Test and Effect Size

Dimension	Benchmark Function	W	Z	p-value	Effect Size (r)	Interpretation
10	Sphere	0	-4.75	<0.001	0.75	Large
10	Schwefel 2.22	2	-4.71	<0.001	0.77	Large
10	Rastrigin	0	-4.78	<0.001	0.75	Large
10	Ackley	1	-4.74	<0.001	0.75	Large
30	Sphere	3	-4.70	<0.001	0.76	Large
30	Schwefel 2.22	4	-4.65	<0.001	0.75	Large
30	Rastrigin	0	-4.78	<0.001	0.75	Large
30	Ackley	2	-4.72	<0.001	0.76	Large
50	Sphere	4	-4.60	<0.001	0.73	Large
50	Schwefel 2.22	5	-4.56	<0.001	0.72	Large
50	Rastrigin	1	-4.75	<0.001	0.76	Large
50	Ackley	6	-4.51	<0.001	0.71	Large

As presented in Table 7. the Wilcoxon signed-rank test shows that the proposed CTM-POA achieves statistically significant performance differences for all tested benchmark functions and dimensions. The comparisons were all $p < 0.001$ which are statistically significant at 5% significance level. Effect sizes ranged from $r = 0.71$ to $r = 0.77$ which are considered large impacts. The continuously large effect sizes indicate that the observed improvement is not only statistically significant but also practically important. The results further show that the performance advantage of CTM-POA is consistent when the problem dimensionality increases from 10 to 50 dimensions.

Table 8. ANN Hyperparameters Before and After CTM-POA Optimization

Hyperparameter	ANN	POA-ANN Optimised Value
Learning rate	0.001	0.0046
Neuron in layer 1	64	96
Neuron in layer 2	32	43
Neuron in layer 3	16	24
Batch size	64	46
Epochs	1000	1000
Initial weight range	Random	POA-optimized
Dropout rate	0.22	1.16
Fitness value	0.064	0.04

This table demonstrates the modification done by CTM-POA on ANN hyperparameters. The parameters made up of three hidden-layer architecture. The learning rate increased gradually from 0.001 to 0.0046, while the hidden neurons increased from 64–32–16 to 96–43–24, providing greater model capacity for learning complex MANET traffic patterns. The batch size was reduced from 64 to 46, and the dropout rate decreased from 0.22 to 0.16 while fitness value reduces from 0.064 to 0.040 Overall, the results indicate that CTM-POA primarily optimized the learning rate, network capacity, batch size, and dropout rate to obtain a more suitable ANN configuration for intrusion detection.

Table 9. Ablation Analysis of CTM-POA-ANN

Model Configuration	POA	Tent Map	ANN Hyperparameter Optimization	Accuracy	Fitness
Conventional ANN	✗	✗	✗	85.41	0.1446
POA-ANN	✓	✗	✓	91.23	0.0735
CTM-POA-ANN	✓	✓	✓	95.82	0.0376

The ablation findings demonstrate a clear increasing improvement from conventional ANN to POA-ANN and finally to CTM-POA-ANN as indicated in Table 9. For conventional ANN, the accuracy was 85.41% with a fitness of 0.1446. The accuracy increased to 91.23% and the fitness decreased to 0.0735 for POA-ANN. The proposed CTM-POA-ANN had the best accuracy of 95.82% and the lowest fitness of 0.0376. This shows that the inclusion of the Chaotic Tent Map in POA yields a further enhancement in the ANN hyperparameter optimization process, confirming the efficiency of the suggested methodology.

Table 10 Sensitivity Analysis

Tent Map Control Parameter (μ)	Accuracy (%)	Fitness $F(\mu)$	Performance
0.3	92.71	0.0525	Moderate
0.4	94.62	0.0532	Good
0.5	95.82	0.0376	Selected/Best
0.6	94.91	0.0409	Good
0.7	93.82	0.0608	Moderate

The sensitivity analysis indicates that the performance of the CTM-POA-ANN is strongly affected by μ . The highest result was found at $\mu = 0.50$ with 95.82% accuracy and the lowest fitness of 0.0376. The performance at $\mu = 0.60$ and $\mu = 0.70$ was slightly lower, which suggests that the optimisation was less effective. Therefore, $\mu = 0.50$ was chosen as the best Tent Map control parameter.

Table 11. ANN Classification Performance

Model	Precision	Recall	Accuracy	F1-Score
ANN	85.61%	85.40%	85.41%	85.35%
POA-ANN	91.44%	90.24%	91.23%	91.14%
CTM-POA-ANN	95.90%	95.84%	95.82%	95.74%

CTM-POA-ANN provides the best classification performance, increasing accuracy from 85.41% for conventional ANN to 95.82%.

Table 12. Comparative Performance of ANN Optimization Algorithms

Model	Accuracy
ANN	85.41%

ABC-ANN	91.04%
GA-ANN	92.20%
PSO-ANN	91.46%
POA-ANN	91.23%
CTM-POA-ANN (Proposed)	95.82%

Table 12 shows the classification accuracies of the different ANN based models. The traditional ANN reached an 85.41% accuracy, which showed the baseline for the intrusion-detection job. GA-ANN was the most accurate among the comparative optimisation strategies with 92.20% accuracy followed by PSO-ANN (91.46%), ABC-ANN (91.04%) and POA-ANN (91.23%). These results showed that the use of metaheuristic optimisation to train ANN may increase the classification performance compared to the traditional ANN. The proposed CTM-POA-ANN achieved the maximum classification accuracy of 95.82% which was better than ANN, ABC-ANN, GA-ANN, PSO-ANN and POA-ANN. The suggested model has an accuracy gain of 10.41 percentage points over the conventional ANN and 4.59 percentage points over the regular POA-ANN. The result shows that integration of Chaotic Tent Map (CTM) into Pelican Optimisation Algorithm improves search process for suitable ANN parameters and leads to better intrusion detection performance. Overall, the findings indicate the efficiency of the proposed CTM-POA-ANN model in enhancing the accuracy of the ANN classification.

CONCLUSION

In this study, a CTM-POA is developed to improve the optimisation ability of the traditional pelican optimisation algorithm. The chaotic tent map technique is used for search process to provide dynamic and irregular search behaviour which enhance the exploration and exploitation balanced. The CTM-POA is assessed on optimisation benchmark functions of varied characteristics: unimodal and multimodal function. The performance is compared with standard POA utilising standard deviation, convergence behaviour, mean, and best fitness. It is recommended to undertake repeated experimental runs and statistical analysis to determine if any apparent improvement in performance is statistically significant. The work provides an algorithmic basis for the application of CTM-POA to increasingly complex optimisation issues. In particular, the suggested approach can be further expanded for optimisation of standard ML models by finding appropriate hyperparameter combinations of Artificial Neural Networks and other prediction systems.

ACKNOWLEDGEMENTS

The authors thank colleagues, researchers, and reviewers for their valuable insights and constructive comments. The support and resources which enabled the research and publication of this work is also appreciated.

REFERENCES

1. Chen, Y., Zhou, T., Chen, Y., Feng, L., Zheng, C., Liu, L., Hu, L., & Pan, B. (2022). HADCNet: Automatic segmentation of COVID-19 infection based on a hybrid attention dense connected network with dilated convolution. *Computers in Biology and Medicine*, 149, 105981. https://pmc.ncbi.nlm.nih.gov/articles/PMC9391231/?utm_source
2. Farshi, T. R. (2021). Battle royale optimization algorithm. *Neural Computing and Applications*, 33(4), 1139–1157. <https://doi.org/10.1007/s00521-020-05004-4>
3. Ghosh, S., Singh, A. and Kumar, S. (2024). HPB3C-3PG algorithm: A new hybrid global optimization algorithm and its application to plant classification. *Ecological Informatics*, 81, 102581. <https://doi.org/10.1016/j.ecoinf.2024.102581>
4. Gopi, S., & Mohapatra, P. (2024). Fast random opposition-based learning Aquila optimization algorithm. *Heliyon*, 10(4), e26187. <https://doi.org/10.1016/j.heliyon.2024.e26187>

5. Liu, Y., Ling, X., Shi, Z., Lv, M., Fang, J., & Zhang, L. (2011). A survey on particle swarm optimization algorithms for multimodal function optimization. *Journal of Software*, 6(12), 2449–2455. https://www.jsoftware.us/show-108-1876-1.html?utm_source=
6. Lu, W., Zhao, H., He, Q., Huang, H., & Jin, X. (2021). Category-consistent deep network learning for accurate vehicle logo recognition. *Neurocomputing*, 463, 623–636. https://www.sciencedirect.com/science/article/pii/S0925231221012145?utm_source=
7. Luo, T., Xie, J., Zhang, B., Zhang, Y., Li, C., & Zhou, J. (2024). An improved levy chaotic particle swarm optimization algorithm for energy-efficient cluster routing scheme in industrial wireless sensor networks. *Expert Systems with Applications*, 241, 122780. https://www.sciencedirect.com/science/article/pii/S0957417423032827?utm_source=
8. Naser, M. Z., Al-Bashiti, M. K., Tapeh, A. T. G., Naser, A., Kodur, V., Hawileh, R., Abdalla, J., Khodadadi, N., Gandomi, A. H., & Eslamlou, A. D. (2025). A review of benchmark and test functions for global optimization algorithms and metaheuristics. *WIREs Computational Statistics*, 17(2), e70028. <https://doi.org/10.1002/wics.70028>
9. Ojo, O. S., Oyediran, M. O., Olagunju, K. M., Opoola, K. B., Ogbonnia, E. O. and Adebisi, A. A. (2025). A hybrid chaotic tent map-pelican optimization algorithm for software defect prediction. *NIPES Journal of Science and Technology Research*, 7(Special Issue: Landmark University International Conference SEB4SDG 2025), 2386–2394. <https://doi.org/10.37933/nipes/7.4.2025.SI282>
10. Suman, A., Gunjan, & Udmale, S. S. (2026). Metaheuristic algorithms: A benchmark-driven functional taxonomy and performance analysis. *Computer Science Review*, 60, 100884. <https://doi.org/10.1016/j.cosrev.2025.100884>
11. Trojovský, P. and Dehghani, M. (2022). Pelican optimization algorithm: A novel nature-inspired algorithm for engineering applications. *Sensors*, 22(3), 1-34 <https://www.mdpi.com/1424-8220/22/3/855/pdf>
12. Varol Altay, E., & Alatas, B. (2020). Bird swarm algorithms with chaotic mapping. *Artificial Intelligence Review*, 53(2), 1373–1414. <https://doi.org/10.1007/s10462-019-09704-9>
13. Wang, L., & Cheng, H. (2019). Pseudo-random number generator based on logistic chaotic system. *Entropy*, 21(10), 960. https://www.mdpi.com/1099-4300/21/10/960?utm_source=
14. Yang, L., and A. Shami. 2020. “On Hyperparameter Optimization of Machine Learning Algorithms: Theory and Practice.” *Neurocomputing* 415: 295–316. <https://doi.org/10.1016/j.neucom.2020.07.061>
15. Zaimoğlu, E. A., Yurtay, N., Demirci, H., & Yurtay, Y. (2023). A binary chaotic horse herd optimization algorithm for feature selection. *Engineering Science and Technology, an International Journal*, 44. <https://doi.org/10.1016/j.jestch.2023.101453>
16. Zou, D., Wang, Z., Zhang, L., Zou, J., Li, Q., Chen, Y., & Sheng, W. (2021). Deep field relation neural network for click-through rate prediction. *Information Sciences*, 577, 128–139. <https://www.sciencedirect.com/science/article/abs/pii/S0020025521006708?utm=>