Journal of Studies in Science and Engineering

EngiScience Publisher

Research Article

Swarm Intelligence Approach for Optimizing Neural Network Architectures in Medical Data Classification

Maher Talal Al-Asaady [1,2], Teh Noranis Mohd Aris [1,*], Nurfadhlina Mohd Sharef [1], Hazlina binti Hamdan [1]

[1] Department of Computer Science, Faculty of Computer Science and Information Technology , Universiti Putra Malaysia, Serdang, 43400 , Malaysia

[2] Department of Network and Computer Software Techniques, Polytechnic College-Mosul, Northern Technical University, Mosul, 41002, Iraq

*Corresponding Author: Teh Noranis Mohd Aris, E-mail: nuranis@upm.edu.my

Article Info

Article History
Received Sep 17, 2025
Revised Jan 21, 2026
Accepted Feb 05, 2026

Keywords
Manta Ray Foraging Optimisation (MRFO)
Multi-layer Perceptron (MLP)
Meta-heuristic Algorithms
Medical Informatics
Neural Network Architecture
Swarm Intelligence

Abstract

The capability of an Artificial Neural Network (ANN) to achieve accurate classification outcomes depends heavily on its architectural design, particularly the number of hidden layers, nodes, and other hyperparameters. Selecting an optimal architecture remains a crucial yet challenging task for improving network accuracy, especially in medical informatics, where reliability is essential. This paper introduces a novel model, MRFO-MLP, which integrates the Manta Ray Foraging Optimization (MRFO) algorithm with a Multi-layer Perceptron (MLP). MRFO is a swarm-intelligence metaheuristic inspired by the cooperative foraging behaviour of manta rays and is employed to explore potential MLP configurations within the solution space. In the MRFO-MLP model, each candidate solution is represented as a binary vector, which reduces both computational time and implementation complexity. Furthermore, a customised fitness function is designed to evaluate candidate architectures by simultaneously considering network complexity and generalisation error, thereby mitigating the risk of overfitting. The proposed model is evaluated using five medical datasets from the UCI repository. Its performance is compared with several optimisation algorithms, including the Genetic Algorithm (GA), Particle Swarm Optimisation (PSO), and traditional machine learning classifiers. Experimental results demonstrate that MRFO-MLP produces efficient and accurate MLP architectures, achieving the best performance on three datasets and the second-best performance on another, highlighting its competitive effectiveness for medical data classification tasks.

1. Introduction

Artificial Neural Networks (ANNs) are mathematical frameworks that mimic the organisation and function of biological neural networks [1]. Within this family of models, the Multi-layer Perceptron (MLP) is a widely used feed-forward network recognised for its suitability for medical data analysis [2]. However,

MLP effectiveness is heavily dependent on the design parameters: the number of hidden layers, the number of nodes per layer, and the activation function [3]. The selection of these parameters is essential in determining classification accuracy, especially in sensitive areas such as medical data classification.

Manually determining the appropriate MLP architecture is a complex, time-consuming process that often results in designs that may lead to overfitting [4]. While few neurons and high generalisation errors characterise underfitting, overfitting is observed in systems with too many layers. Therefore, one of the most important subjects of study is the automatic choice of the optimal MLP configurations.

Since hyperparameters are not learned from data, they must be manually adjusted to ensure the model gives good results [5]. Regarding MLP performance, specific hyperparameters, such as the number of hidden layers, the number of nodes per layer, the choice of activation functions, and the number of epochs, are important. The number of hidden layers and nodes in each hidden layer enables the network to learn complex patterns, but may lead to overfitting if not well-regulated [6]. The activation function introduces nonlinearity into the network, which is essential for modelling complex nonlinear functions. The most popular activation functions include Sigmoid, Rectified Linear Unit (ReLU), and the Hyperbolic tangent function [7].

Several disciplines, including ANN optimisation, are now increasingly using Meta-Heuristic Algorithms (MHAs) [8]. They use ideas from nature or living organisms to help find near-optimal solutions in areas with many possible paths [9]. Various works in the literature have applied several MHAs, including the Genetic Algorithm (GA) [10], Particle Swarm Optimisation (PSO) [11], and the Grey Wolf Optimisation (GWO) [12], to enhance the design of ANNs. While these algorithms deliver good results, there is still room to improve the optimisation of ANN architectures.

This study proposes a new model, MRFO-MLP, in which the Manta Ray Foraging Optimisation (MRFO) algorithm improves the architecture of an MLP neural network. The approach mimics the feeding habits and group behaviour of manta rays to explore a suitable MLP architecture that improves the accuracy of medical data classification. This proposed method uses a customised fitness function that aims to avoid heavy networks by balancing generalisation error and the total number of network links.

2. Literature Review

Swarm intelligence has experienced impressive growth over the last few years, with many new bio-inspired algorithms being developed to address more complex optimisation problems. Recent work has

seen the Mountain Gazelle Optimiser [13], Artificial Rabbits Optimisation [14], Coyote Optimisation Algorithm [15], Sand Cat Swarm Optimisation [16], and the Multi-strategy Chimp Optimisation Algorithm [17], among others, all of which have shown the ongoing advance towards the ability to mimic natural behaviours to solve computational problems.

Moreover, the usefulness of these MHAs has expanded well beyond hypothetical function optimisation into the real world and important sectors. Other extensive literature reviews and research papers have emphasised their crucial role in image processing and computer vision applications [18], efficient decentralised IoT network layouts [19], ideal task scheduling in fog-cloud settings [20], medical data classification [5], and innovative traffic recovery models grounded on context-aware reinforcement learning [21]. All these studies highlight the strength of meta-heuristic models in addressing high-dimensional, dynamical engineering problems.

Various strategies have been proposed to optimise ANN architecture. Previous studies have employed constructive and pruning techniques [22] in which architectures are developed sequentially by adding or removing neurons and connections. However, these methods are often susceptible to local optima and may not fully leverage the entire architectural space [3]. Some optimisation techniques, including Bayesian optimization [23] and random search [24], have been employed to improve the design and hyperparameters of MLP neural networks.

In addition, using MHAs has become more popular in streamlining ANN architectures logically and responsively. These techniques are architectural parameters (e.g., number of hidden neurons, activation functions) within the encoded solution spaces whose exploration is used to discover optimal settings. In contrast to a manual trial-and-error approach, MHAs allow worldwide searching, are scalable, and are adaptable to various problem areas [25].

GA was successfully applied to optimise ANN architectures via an iterative population-based search over the search space. El-Hassani et al. [10] proposed a real-coded GA to optimise the MLP architecture's parameters. Likewise, Arroyo and Delima [26] used GA to optimise design parameters to predict cardiovascular disease, and Domashova et al. [27] used GA to identify optimal ANN architectures to detect fraud. Lima et al. [28] used GA to classify biosignals, thereby improving the accuracy of detecting wrist orientation, muscle contraction, and voice parameters.

PSO has also been widely applied to the optimisation of ANN architectures. Cheng et al. [29] have

proposed Lifespan-PSO, which uses a lifespan mechanism to optimise ANN architectures for classifying coal mining images. Sarkar et al. [30]applied PSO to the electromagnetic inverse problem, while Jamous et al. [31] applied PSO to optimise the ANN architecture for the classification of chemical sensor data.

Other MHAs have also been adopted. Han et al. [32] proposed SaDENAS, a self-adaptive DE approach that stabilises the search for optimal architectures. Moradi et al. [33] present the optimisation of an MLP architecture for modelling the Equal Channel Angular Pressing-Conform (ECAP-C) process using PSO and WOA. Simsek and Alagoz [34] studied the use of GWO to optimise an ANN for air quality monitoring applications. In Bansal et al. [3], the Lion Optimisation Algorithm (LOA) was utilised to improve MLP architectures for classification tasks. Ghanou and Bencheikh [35] used Ant Colony Optimisation (ACO) to optimise ANN architectures.

Table 1. Overview of studies utilising MHAs for ANN architecture optimisation

Ref	MHA	Method	Dataset/s	Key findings
[10]	GA	Hyperparameter tuning, including architecture optimisation	Wine, Hypothyroid, Iris, Cancer	Enhanced MLP performance through real-coded GA
[26]	GA	Optimisation of MLP configurations	Cardiovascular dataset	Improved accuracy in disease prediction
[27]	GA	ANN architecture selection for fraud detection	Banking transaction data	ANN design has been identified for detecting fraudulent transactions
[28]	GA	MLP design optimisation for bio-signal classification	EMG signals, voice parameters	Improved accuracy in biosignal classification
[29]	PSO	Lifespan-based PSO for neural architecture search (NAS)	Multiple real coal mine scene image datasets	Achieved higher accuracy and reduced complexity
[30]	IPSO	Hyperparameter optimisation for inverse EM modelling	EBG and UWB antenna datasets	Outperformed GA and conventional PSO
[31]	PSO	Topology optimisation	Chemical sensors dataset	PSO improved classification performance
[32]	DE	Architecture search and training optimisation	CIFAR-10 CIFAR-100	Compact MLPs with competitive accuracy
[33]	PSO-WOA	MLP optimisation for ECAP-C modeling	ECAP-C experimental data	Accurate modeling via PSO-WOA
[34]	GWO	ANN optimisation for the e-nose system	Air quality monitoring data	Enhanced gas classification accuracy
[3]	LOA	Architecture optimisation for classification tasks	10 UCI datasets	Improved generalisation
[35]	ACO	ANN architecture optimisation	Iris, Cancer, Seeds	Discovered optimised MLP architectures

Further improvements to the hybrid, innovative optimisation strategies have led to better ANN performance. A hybrid PSO–GWO approach is proposed by Aviles et al. [36] for the optimisation of MLP

hyperparameters in the Electromyography (EMG) signal classification case study. Sambatti et al. [37] used the Multi-Particle Collision Algorithm (MPCA) to develop solutions to multiple objectives optimisation of ANN accuracy and complexity.

Table 1 summarises various studies that employ MHAs to optimise ANN architectures. Each entry includes the algorithm, the optimisation method, the datasets, and the key findings.

Further improvements to the hybrid, innovative optimisation strategies have led to better ANN performance. A hybrid PSO–GWO approach is proposed by Aviles et al. [36] for the optimisation of MLP hyperparameters in the Electromyography (EMG) signal classification case study. Sambatti et al. [37] used the Multi-Particle Collision Algorithm (MPCA) to develop solutions to multiple objectives optimisation of ANN accuracy and complexity.

Table 1 summarises various studies that employ MHAs to optimise ANN architectures. Each entry includes the algorithm, the optimisation method, the datasets, and the key findings.

Even though there is much evidence for the successful application of MHAs to optimise ANN architectures, the No-Free-Lunch (NFL) theorem [38] stipulates that no optimisation method globally outperforms all others on all problems. This is why continued research into developing new techniques and improving existing ones, tailored to specific applications, is imperative. In this study, the proposed MRFO-MLP model supports this continuous process by employing MRFO to enable the MLP model to evolve its structure and achieve high classification accuracies.

3. Materials and Methods

3.1. Multi-layer Perceptron (MLP)

An MLP is a type of ANN widely used for supervised learning in classification and regression tasks. It adopts a feed-forward architecture composed of multiple layers of interconnected neurons arranged in parallel. Each neuron processes input values weighted by adjustable parameters and applies a nonlinear activation function to produce its output. The Backpropagation (BP) method is often used to iteratively optimise these weights and minimise the error between the network's predicted and actual values using a given training dataset [36].

An MLP neural network consists of three main parts: the input layer, the hidden layers, and the output layer. The input layer receives information, sends it to the hidden layer for processing, and finally, the

output layer gets the results. The output layer produces the classification outcome. While weights determine how strongly neurons are connected, every hidden and output neuron has a bias term that allows it to refine its response and contribute to an accurate prediction [39]. The structure of the MLP is shown in Figure 1.

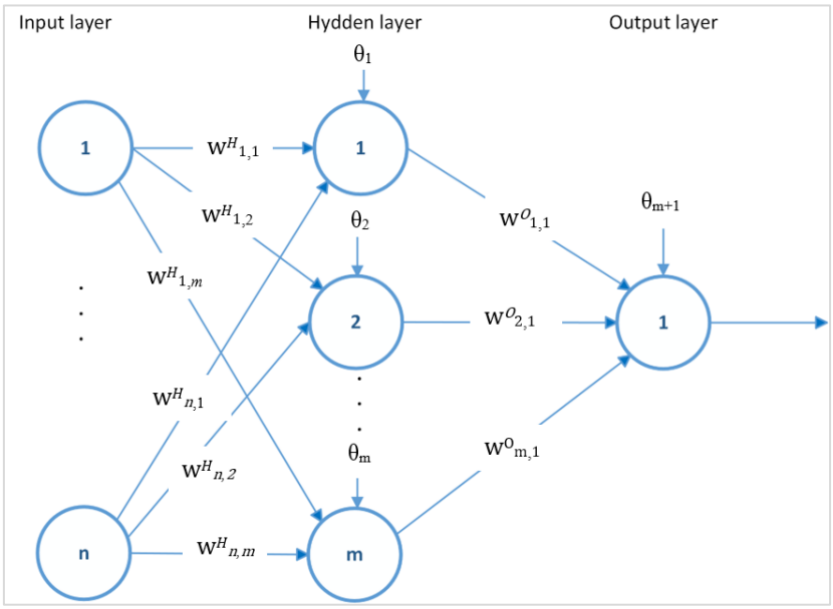


Figure 1. Sample of MLP Architecture

Neurons perform activation and summation in hidden layers. As seen in Eq. (1), summation computes the weighted outputs from the neurons in the preceding layer, then adds a bias term.

$$Sum_j^H = \sum_{i=1}^{n} w_{i,j}^H \times X_i + \theta_j \quad (1)$$

where $w^H{}_{i,j}$ is the weight from input neuron i to hidden neuron j. X_i is the output of the input neuron i, and θ_j is the bias of the neuron j.

The activation function, such as the sigmoid function shown in Eq. (2), determines neuron activation by considering the weighted sum and bias. Where $y^H{}_j$ is the last output of the hidden neuron j.

$$y_j^H = f(Sum_j^H) = \frac{1}{1+e^{-Sum_j^H}} \quad (2)$$

The summation for an output neuron is shown in Eq. (3), leading to the final output via activation:

$$Sum^o = \sum_{j=1}^{m} w_{i,j}^O \times y_j^H + \theta_{m+1} \quad (3)$$

$$O = f(sum^o) = \frac{1}{1+e^{-sum^o}} \quad (4)$$

Because the MLP is used for classification tasks, its output must represent a discrete value that accurately distinguishes between classes. In this study, we implemented binary classification on medical

datasets. To convert the continuous output into a definitive binary output Y', which distinguishes between the two classes using a threshold T as outlined in Eq. (5).

$$Y' = \begin{cases} 0, \ if \ \ O \ < \mathrm{T} \\ 1, \ if \ \ \mathrm{O} \geq \mathrm{T} \end{cases} \tag{5}$$

3.2. Manta Ray Foraging Optimisation (MRFO)

Manta Ray Foraging Optimisation (MRFO), proposed by Zhao et al. [40], is a meta-heuristic algorithm based on the advanced foraging processes of manta rays. These sea creatures have flat bodies with large, wing-like pectoral fins and unique cephalic lobes, which are horn-like and are used to feed. During foraging, manta rays utilise their cephalic lobes to funnel water and prey into their mouths. Figure 2 illustrates a real manta ray in the ocean (A) and a labelled diagram of its anatomical features (B), including the cephalic lobe, mouth, eye, pectoral fin, and tail, integral to its unique foraging methods.

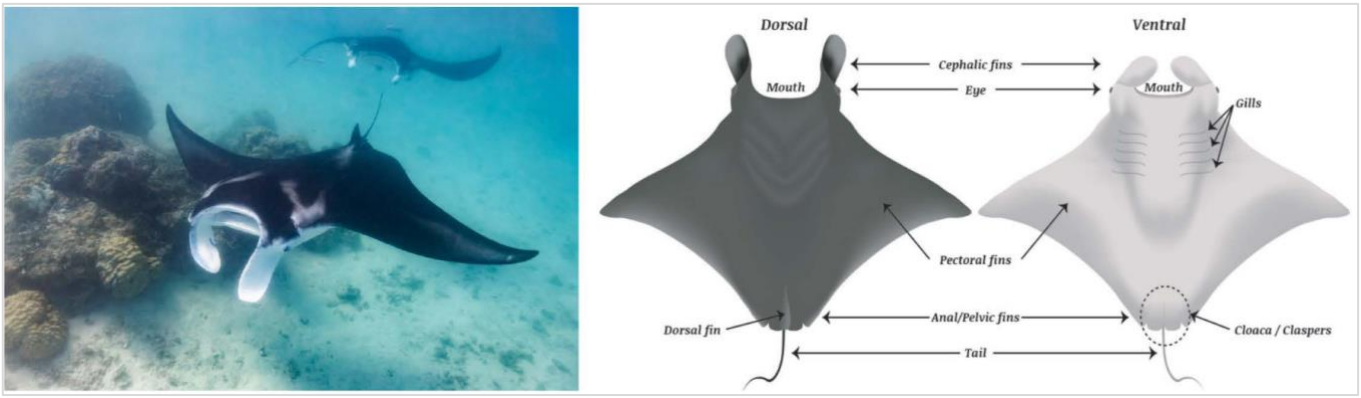


Figure 2. (A) Manta ray in the ocean; (B) Schematic illustration of the manta ray showing key anatomical features used in foraging [41]

The MRFO algorithm emulates three main foraging strategies observed in manta rays: Chain foraging, cyclone foraging, and somersault foraging.

3.2.1. Chain Foraging

Manta rays use cooperative feeding in the chain foraging strategy. They are formed linearly and can have over 50 members. All the individuals match up behind each other, ensuring that those behind can still get plankton that the front rays missed. This sequential positioning increases overall foraging efficiency and minimises the risk of prey being overlooked in crowded feeding areas.

At the given stage of MRFO, every manta ray (solution) moves its position depending on the position of the leading ray, the solution that was the most successful so far, in comparison to the previous ray. This can be represented as in Eq. (6).

$$x_i^d(t+1) = \begin{cases} x_i^d(t) + r.(x_{best}^d(t) - x_i^d(t)) + \alpha.(x_{best}^d(t) - x_i^d(t)) & i = 1 \\ x_i^d(t) + r.(x_{i-1}^d(t) - x_i^d(t)) + \alpha.(x_{best}^d(t) - x_i^d(t)) & i = 2, \ldots, N \end{cases} \quad (6)$$

where $x_i^d(t)$ represents the location of the i^{th} individual at time t in the d^{th} dimension. $x_{best}^d(t)$ is the position of the optimal solution identified in the process. r is a random number within the range [0, 1]. α is a weight coefficient defined with the following formula in Eq. (7).

$$\alpha = 2.r.\sqrt{|log(r)|} \quad (7)$$

3.2.2. Cyclone Foraging

Manta rays use the foraging behaviour of the cyclone when they sense the presence of a high level of prey concentration. They form a spiral shape, combining their heads and tails into a vortex that sweeps plankton into the mouth. This spiralling movement generates upward currents and directly delivers plankton-rich water to the former, indicating an advanced exploitation method. This behaviour is modelled in MRFO, as shown in Eq. (8).

$$x_i^d(t+1) = \begin{cases} x_{best}^d + r.(x_{best}^d(t) - x_i^d(t)) + \beta.(x_{best}^d(t) - x_i^d(t)) & i = 1 \\ x_{best}^d + r.(x_{i-1}^d(t) - x_i^d(t)) + \beta.(x_{best}^d(t) - x_i^d(t)) & i = 2, \ldots, N \end{cases} \quad (8)$$

$$\beta = 2e^{r_1 \frac{T-t+1}{T}}.sin(2\pi r_1) \quad (9)$$

where T is the maximum number of repetitions, r_1 is a random number uniformly generated in the range [0,1], and β is a weight coefficient.

Another method to improve the global search of the model is to choose a new random point as the reference point for the entire search space, as shown in Eq. (10). Forcing each individual to locate a different position far from the current optimal position can be expressed as in Eq. (11).

$$x_{rand}^d = L_b^d + r.(U_b^d - UL_b^d) \quad (10)$$

$$x_i^d(t+1) = \begin{cases} x_{rand}^d(t) + r.(x_{rand}^d(t) - x_i^d(t)) + \beta.(x_{rand}^d(t) - x_i^d(t)) & i = 1 \\ x_{rand}^d(t) + r.(x_{i-1}^d(t) - x_i^d(t)) + \beta.(x_{rand}^d(t) - x_i^d(t)) & i = 2, \ldots, N \end{cases} \quad (11)$$

where $x_{rand}^d(t)$ is the random location of random production, $U_b^d - L_b^d$ are the lower and upper bounds of the search space, respectively.

3.2.3. Somersault Foraging

Such an action implies that manta rays must be observed flipping back and forth around plankton

clusters in the sea. The movement that occurs around and around results in local water movement, pulling plankton towards the surface. It is best when plankton are dispersed, as this appears to be a more efficient way for them to access more food. In the MRFO algorithm, every manta ray somersaults about the optimal solution with a random somersault factor. This behaviour is modelled as in Eq. (12).

$$x_i^d(t+1) = x_i^d(t) + S.(r_2.x_{best}^d - r_3.x_i^d(t)) \quad i = 1,\ldots,N \tag{12}$$

where S is the somersault factor, set to $S=2$. r_2 and r_3 are random numbers in [0,1].

According to these biological concepts, MRFO provides consistent output when searching large, complex search spaces and seeking near-optimal solutions, balancing global and local search. These three foraging behaviours are simulated by the MRFO shown in Figure 3 [42].

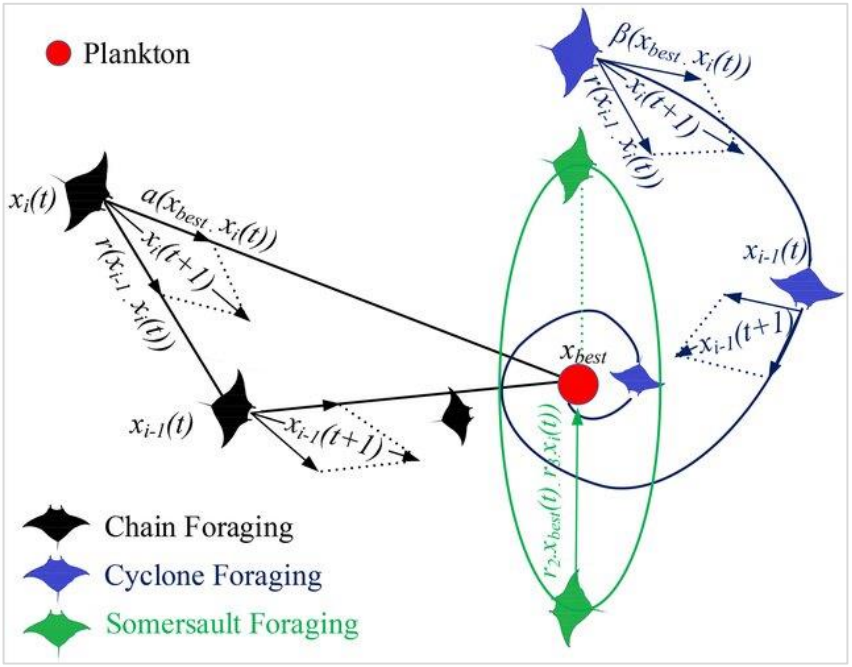


Figure 3. Manta ray foraging strategy

4. Proposed Model: MRFO-MLP

In the MRFO-MLP model, the MRFO algorithm optimises the MLP architecture by determining the number of neurons in one or two hidden layers based on the dataset and the problem's complexity. Each candidate architecture is binary-encoded, with a fixed-length binary vector representing each hidden layer.

MRFO searches this binary-encoded space to determine architectures with structural efficiency and functional effectiveness. The fitness of each architecture is measured using a hybrid fitness function that combines training Mean Squared Error (MSE) and model complexity (number of parameters and iterations). This architecture optimisation strategy enables the model to:

- Avoid manual tuning of network architecture,
- Explore both shallow and deeper MLP topologies,
- Adapt the complexity of the network to the characteristics of the dataset,
- Balance model performance and simplicity through regularised fitness evaluation.

The concept of MRFO-based architectural optimisation is illustrated in Figure 4, which shows the interaction between the MLP model and the MRFO algorithm during the architecture search phase.

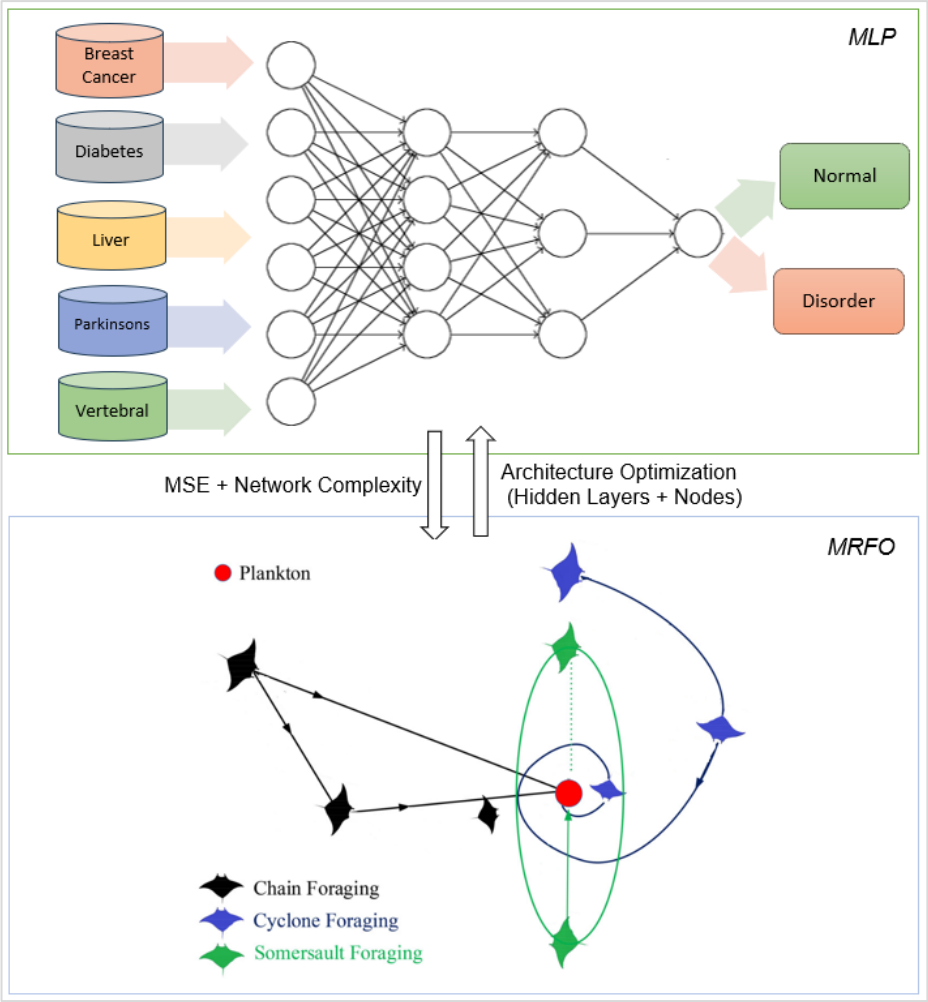


Figure 4. Conceptual overview of MLP architecture optimisation using the MRFO algorithm

4.1. Solution Representation

In MRFO-MLP, each solution' manta ray' is represented by a binary string as per the corresponding MLP node candidate solution in the MLP architecture. In this work, MLPs with two hidden layers are designed, and the binary string representing the data is partitioned into three parts for the number of nodes and the activation function in each hidden layer. Table 2 illustrates the range of values that can be used to represent network architecture in the MRFO-MLP model.

Table 2. Range and types of design parameters

Parameter	# Bits	Range/Values
# Hidden nodes 1st	5	5-31
# Hidden nodes 2nd	5	0-31
Activation function	2	[00-Sigmoid, 01-Hyperbolic Tangent (Tanh), 10-ReLU, 11-reserved]

The binary encoding improves the implementation process by allowing the model to perform bitwise operations and enhances the network's convergence speed. It eliminates rounding and boundary issues in real values, making this encoding suitable for discrete architecture optimization [3, 43]. The MRFO-MLP model uses a 12-bit binary string. The first five bits (0-4) are related to the first hidden layer, the other five bits (5-9) correspond to the second hidden layer, and the last two bits (10-11) represent the activation function. Figure 5 illustrates the solution binary string for the parameters. When the optimised MLP has only one hidden layer, the bit string representing the number of neurons in the second hidden layer is '00000'. The lower bound for the first hidden layer was set to 5 to avoid degenerate, underparameterised networks and to keep the architecture search space manageable. $H_1 \in [5,31]$, $H_2 \in [0,31]$, and $H_2 = 0$ indicates a single hidden layer.

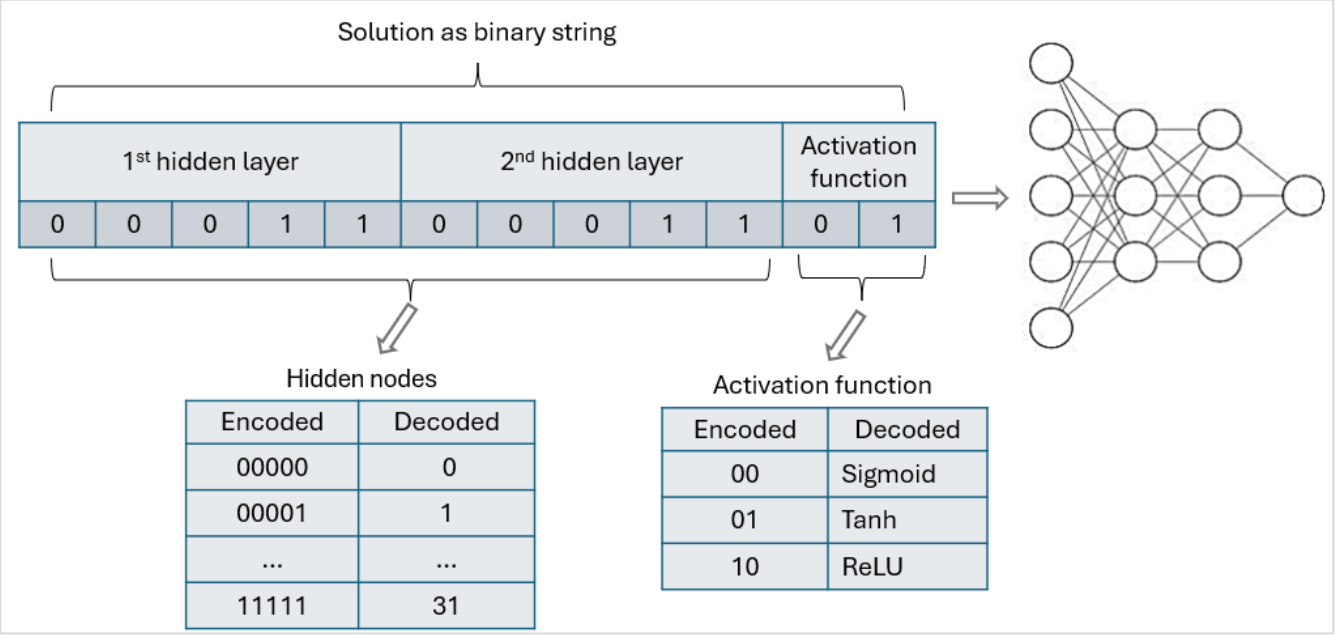


Figure 5. The representation of the solution binary string for the parameters

4.2. Fitness Function

Every 'Manta Ray' in the algorithm is a solution in the binary solution space at any given time. Throughout the search process, manta rays update their positions using different foraging strategies. Each position corresponds to a set of MLP architecture parameters. The MLP is then trained and optimised with

these parameters using the BP method. A fitness function evaluates each position explored by a 'Manta ray' by assessing the MLP's performance on validation data.

To avoid overfitting, the training procedure is terminated by early stopping, as recommended by Bishop [44]. During early stopping, the MLP is evaluated using the validation set after every ten training epochs. The final fitness score is derived from the Mean Squared Error (MSE) of the validation set combined with a complexity penalty. After training, the fitness of the trained MLP is determined by calculating the MSE and applying a penalty factor, as shown in Eq. (13).

$$fitness(MLP) = MSE + Penality \tag{13}$$

where MSE is used to assess the network model's efficiency by measuring the average of the squares of the errors, as shown in Eq. (14).

$$MSE = \frac{1}{n} \sum_{k=1}^{n} (y_k - \acute{y_k})^2 \tag{14}$$

Here, n is the total number of training samples, y'^k is the predicted output for the k^{th} sample, and y_k is the actual binary target. Although MSE is typically used in regression problems, in classification problems, it highlights the difference between the predicted continuous value and the expected binary result [45, 46].

The 'Penalty' term is included to promote simpler, more efficient MLP architectures, as shown in Eq. (15). This penalty comprises two factors: the total number of weights and biases (*con*) and the number of training iterations (*itr*). The first term penalises overly complex networks, while the second discourages architectures that require excessive training epochs to converge.

$$Penality = \alpha \cdot con + \beta \cdot itr \tag{15}$$

The coefficients α=5×10^{-7} and β=5×10^{-6} were selected empirically through a grid search approach. Multiple combinations were tested, and these values consistently produced MLP architectures that achieved a favourable balance between classification accuracy and model complexity across various datasets.

4.3. Optimisation Process in MRFO-MLP

In the proposed MRFO-MLP model, MRFO is utilised as an outer-loop optimiser to explore the space of MLP architectures, and backpropagation is used as an inner-loop optimiser to train network weights. The number of hidden neurons, the number of hidden layers, and the activation function are coded in binary in each MRFO candidate solution. To each candidate, an MLP is trained, evaluated, and constructed, and its performance is returned to MRFO as a fitness value. This high-ranking feedback loop enables MRFO to

repeatedly focus the search on efficient and effective multi-layer model structures.

The formulation of improving the MLP architecture with the MRFO-MLP commences with a randomly instantiated population of manta rays (MLP_i) within the binary search domain (MLP_u) and (MLP_l). This initial population is encoded into the MLP's architecture and trained using BP with input training instances from the medical dataset. At the end of the training, the network is tested on a test dataset, where the computed output is compared with the target output. The difference between these outputs is measured using MSE as in Eq. (14). The fitness is calculated, and the optimum fitness is attributed to (MLP_{best}).

Algorithm 1. The MRFO-MLP model

1: **Initialize #** iterations (T_{max})**, # population (*N*), upper and lower bounds (MLP_u) and (MLP_l).**
2: Generate random initial populations (MLP_i**) within bounds.**
3: Update MLP_{best} with the architecture having the lowest fitness.
4: WHILE stopping condition not met **DO**
5: **FOR *i*=1 to *N* DO**
6: **Train** MLP_i **using BP based on training set, and evaluate it based in validation set.**
7: **Evaluate the fitness value (MSE+Penalty) for each** MLP_i**.**
8: **IF** $f(MLP_i) < f(MLP_{best})$ **THEN** Replace MLP_{best} with MLP_i
9: **IF** rand < 0.5 **THEN** // *Cyclone Foraging*
10: **IF** t/T_{max} < rand **THEN**
11: Update each MLP_i architecture using Eqs. 10,11.
12: **ELSE**
13: Update each MLP_i architecture using Eq. 8.
14: **END IF**
15: **ELSE**
16: Update each MLP_i architecture using Eq. 6. // *Chain Foraging*
17: **END IF**
18: **Train** MLP_i **using BP based on training set, and evaluate it based in validation set.**
19: **Evaluate the fitness value (MSE+Penalty) for each** MLP_i**.**
20: **IF** $f(MLP_i) < f(MLP_{best})$ **THEN** Replace MLP_{best} with MLP_i
21: Update each MLP_i architecture using Eq. 12. // *Somersault Foraging*
22: **END FOR**
23: **END WHILE**
24: **Output** the best MLP architecture parameters MLP_{best}

The MRFO algorithm iteratively updates the positions of the manta rays to represent new MLP architectures (MLP_i). During these updates, the MLP is trained with various architectures, and the fitness value is recalculated based on the MSE and penalty values. This process is repeated until the optimal solution is achieved or until a specified termination condition is met, aiming to determine the most efficient MLP configuration for medical data classification. The solution with the lowest fitness value, which corresponds to the minimised MSE plus the penalty factor, is selected as the optimal configuration for future classification tasks. Algorithm 1 presents the pseudo-code for the MRFO-MLP model, and Figure 6 illustrates the MRFO-MLP process flowchart for optimising the MLP architecture.

5. Results and Discussions

The MRFO-MLP model is evaluated on five medical datasets obtained from The UCI Machine Learning Repository [47]. The datasets pertain to five medical issues, each focused on identifying a specific disease. These are Breast cancer, Diabetes, Liver, Parkinsons, and Vertebral. A brief description of all the datasets is given in Table 3. The datasets were split into 66% for training, 17% for validation, and 17% for testing. The training set was used to learn the network weights, while the validation set was employed within the MRFO optimisation loop to calculate fitness and trigger early stopping. The test set was held out completely during the optimisation process and used only for the final performance assessment. A stratified sampling technique was used to ensure the subset remained proportional to the original data [48].

Moreover, the max-min normalisation procedure was used to scale all attribute values, as shown in Eq. (16).

$$X'_i = \frac{X_i - min_X}{max_X - min_X} \tag{16}$$

where X_i is the original value of feature X for instance i and X'_i is the normalised value. The min_X and max_X are the lower and upper bounds of feature X_i.

Table 3.Medical Datasets

Dataset	Features	Instances	Label 1	Label 2
Breast cancer	9	699	Benign (458)	Malignant (241)
Diabetes	8	768	Diabetes (268)	Normal (500)
Liver	6	345	Infected (200)	Normal (145)
Parkinsons	22	195	Disorder (147)	Normal (48)
Vertebral	6	310	Disorder (210)	Normal (100)

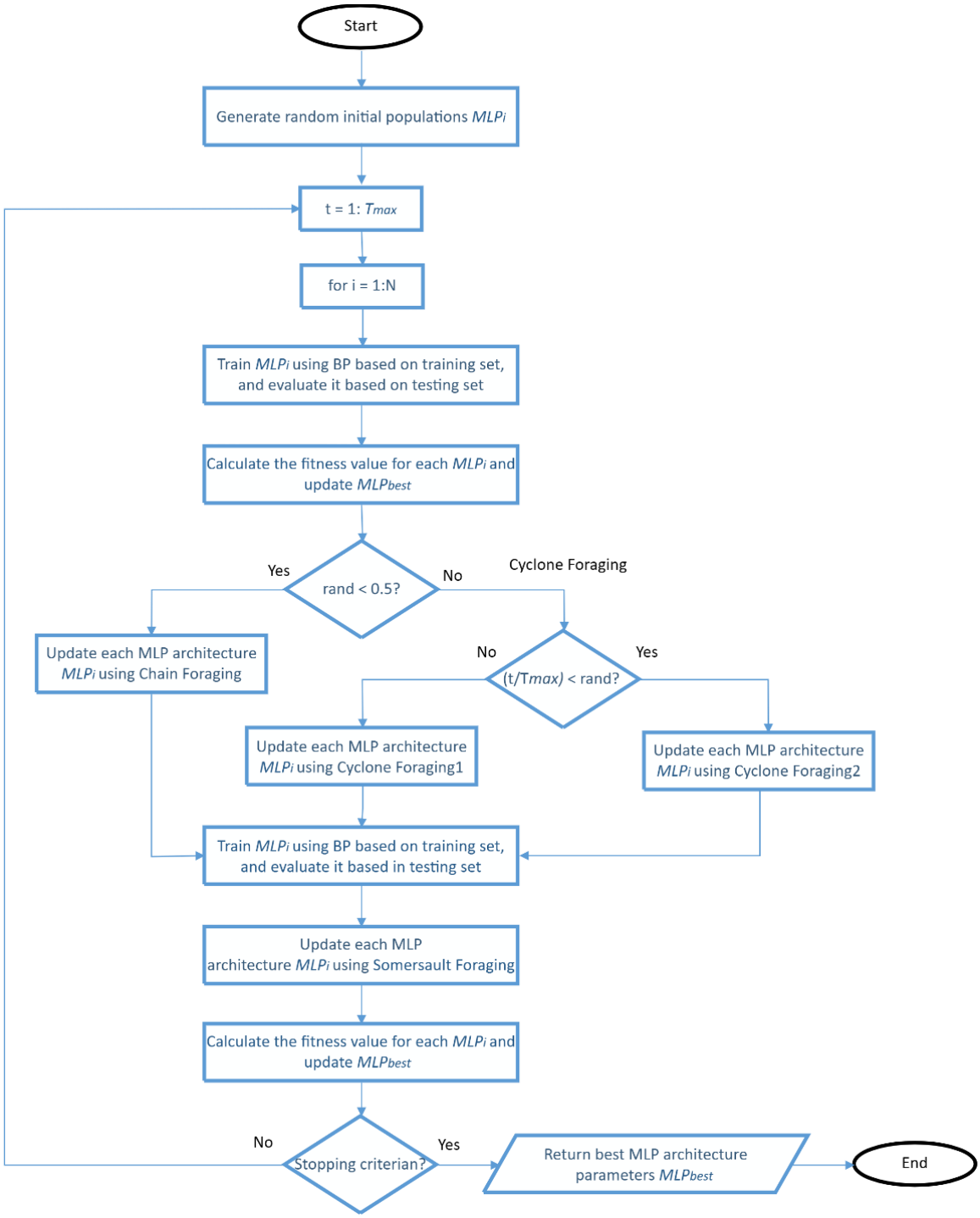


Figure 6. Conceptual overview of the MRFO-MLP model

This study compared the MRFO-MLP model with other state-of-the-art algorithms that optimise the MLP neural network architecture, including GA, PSO, and GWO. All competing algorithms (MRFO-MLP, GA-MLP, and PSO-MLP) were run under the same experimental conditions to ensure a fair, reproducible comparison. The population size was set to N = 50, and the maximum allowed was 100 iterations across all algorithms. The termination test was set at the maximum number of iterations. Moreover, to reduce the impact of the stochastic randomness, 30 independent runs of each algorithm were used on each dataset. All

the experiments were carried out on the same hardware to rule out any performance differences caused by differences in computational resources. The parameter settings of each algorithm are illustrated in Table 4 based on previous studies [3, 49].

Table 4. Algorithm parameter settings

Algorithm	Parameter	Value
MRFO	Somersault foraging	1
GA	Probability of crossover	0.8
	Probability of mutation	0.05
	Probability of selection	0.5
	Selection type	Roulette wheel
PSO	Inertia weight	0.7
	Cognitive component	1.1
	Social constant	1.7

To analyse the effect of population size (N) on the accuracy of classification, and to test the strength of the offered method. Sensitivity analysis across all datasets was performed by testing a range of population sizes (N). The empirical findings showed that population decline led to inadequate exploration, resulting in a worsening of average accuracy. On the other hand, the population size increase showed no significant improvement in accuracy but did increase computational runtime.

Table 5. Best MLP architectures

Dataset	Model	Architecture #I × #H1 × #H2	Activation Function
Breast cancer	GA	9 × 6 × 3	Tanh
	PSO	9 × 7 × 0	Tanh
	MRFO-MLP	9 × 7 × 15	ReLU
Diabetes	GA	8 × 16 × 1	Sigmoid
	PSO	8 × 29 × 8	Tanh
	MRFO-MLP	8 × 22 × 10	Tanh
Liver	GA	6 × 16 × 25	ReLU
	PSO	6 × 22 × 12	Tanh
	MRFO-MLP	6 × 14 × 10	Sigmoid
Parkinsons	GA	22 × 6 × 12	ReLU
	PSO	22 × 29 × 2	Tanh
	MRFO-MLP	22 × 19 × 6	Tanh
Vertebral	GA	6 × 17 × 4	Sigmoid
	PSO	6 × 27 × 11	Tanh
	MRFO-MLP	6 × 25 × 27	Tanh

The proposed MRFO-MLP model was implemented in Python using the open-source library Optimis [50]. The MLP model was created using the Scikit-Learn library. The experiments were conducted on an Intel i7 2.11 GHz processor (8 cores, 16 GB RAM). The model is trained using early stopping and a batch size of 100 in the BP method. Each experimental run was initiated with randomised initial parameters and repeated 30 times across independent runs. The MHAs comprised 50 individuals, and each algorithm performed 100 iterations.

Table 5 presents the specifications of the optimal MLP architectures, including the number of inputs (#I), the number of nodes in the first hidden layer (#H1), and the number of nodes in the second hidden layer (#H2). These specifications were obtained using the MRFO-MLP, GA, and PSO algorithms across all five medical datasets, based on their respective test accuracies.

Table 6. Test the accuracy of all algorithms

Dataset/Model	NB	DT	RF	SVM	GA	PSO	MRFO-MLP
Breast cancer							
AVG	0.9611	0.9375	0.9665	*0.9675*	0.9542	0.9625	**0.9716**
STD	0.0089	0.0143	0.0096	0.0092	0.0064	0.0088	0.0028
Best	0.9832	0.9664	0.9916	0.9874	0.9731	0.9715	0.9895
Diabetes							
AVG	0.7518	0.7094	0.7589	*0.7704*	0.7621	0.7678	**0.7811**
STD	0.0222	0.0298	0.0236	0.0174	0.0072	0.0336	0.0071
Best	0.7939	0.7595	0.8053	0.8015	0.7713	0.7849	0.7938
Liver							
AVG	0.5551	0.6387	**0.7164**	0.5757	0.6862	0.6692	*0.6954*
STD	0.0602	0.0453	0.0359	0.0021	0.0023	0.0074	0.0213
Best	0.6780	0.7542	0.7712	0.5763	0.6943	0.6853	0.7235
Parkinsons							
AVG	0.6955	0.8373	**0.8861**	0.8572	*0.8723*	0.8625	0.8665
STD	0.0490	0.0352	0.0388	0.0258	0.0132	0.0154	0.0200
Best	0.7612	0.8955	0.9552	0.8955	0.8792	0.8752	0.8707
Vertebral							
AVG	0.7733	0.7953	*0.8384*	0.7604	0.7842	0.8314	**0.8397**
STD	0.0324	0.0359	0.0301	0.0315	0.0112	0.0153	0.0281
Best	0.8302	0.8679	0.8774	0.8208	0.7983	0.8431	0.8535

For the comparison, we obtained the average (AVG), standard deviation (STD), and best test accuracy of the 30 runs for all models. We also included results from existing machine learning classifiers, including Naïve Bayes (NB), Decision Tree (DT), Random Forest (RF), and Support Vector Machine (SVM). The findings are presented in Table 6. The top results are bolded, and the second-place results are italicised.

In the Breast Cancer dataset, the MRFO-MLP achieved the highest average accuracy of 0.9716, which is significantly higher than that of all the other models. The STD for MRFO-MLP was also the lowest at 0.0028, showing that the algorithm was stable in its performance even when run multiple times. For the Diabetes dataset, MRFO-MLP again performed best, achieving an average accuracy of 0.7811. The STD was also relatively low at 0.0071. The results on the Liver dataset differed from those in the previous ones. Here, RF had the highest average accuracy of 0.7164, while MRFO-MLP was closely behind with an average accuracy of 0.6954 and a best accuracy of 0.7235. In the Parkinson's dataset, RF achieved the highest accuracy among all models, with an average of 0.8861. MRFO-MLP achieved an average accuracy of 0.8665, which is quite close to RF but higher than that of the other models. Finally, the MRFO-MLP model achieved the highest mean accuracy of 0.8397 for the vertebral dataset, surpassing all the other models.

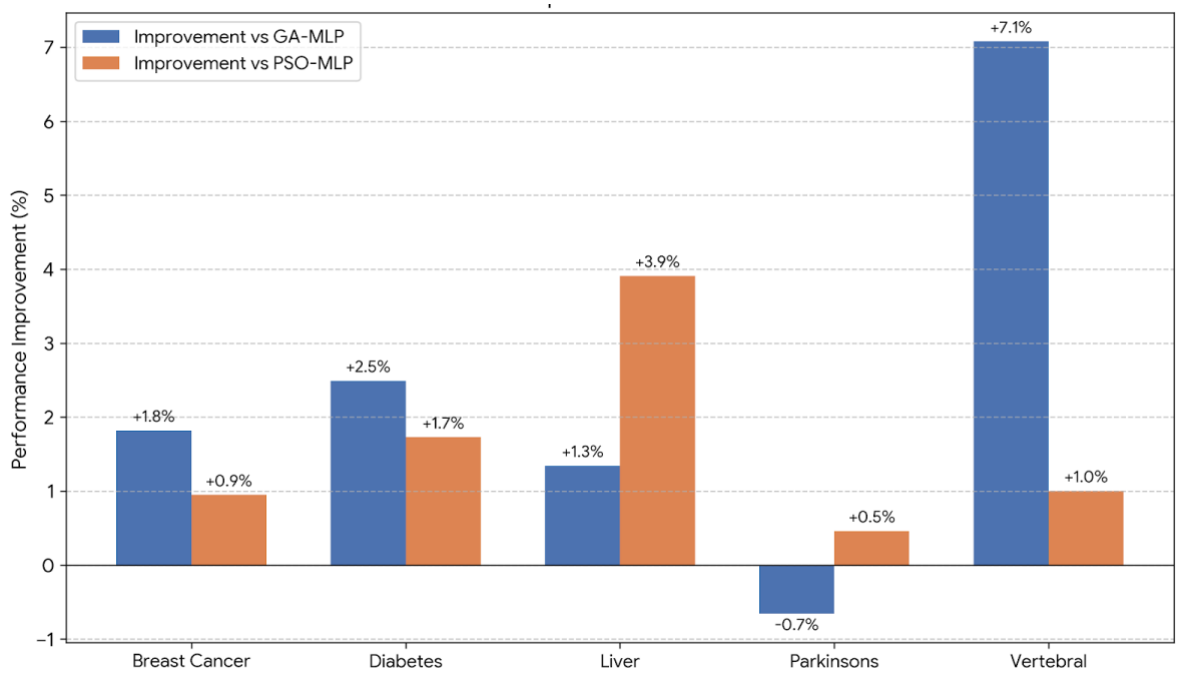
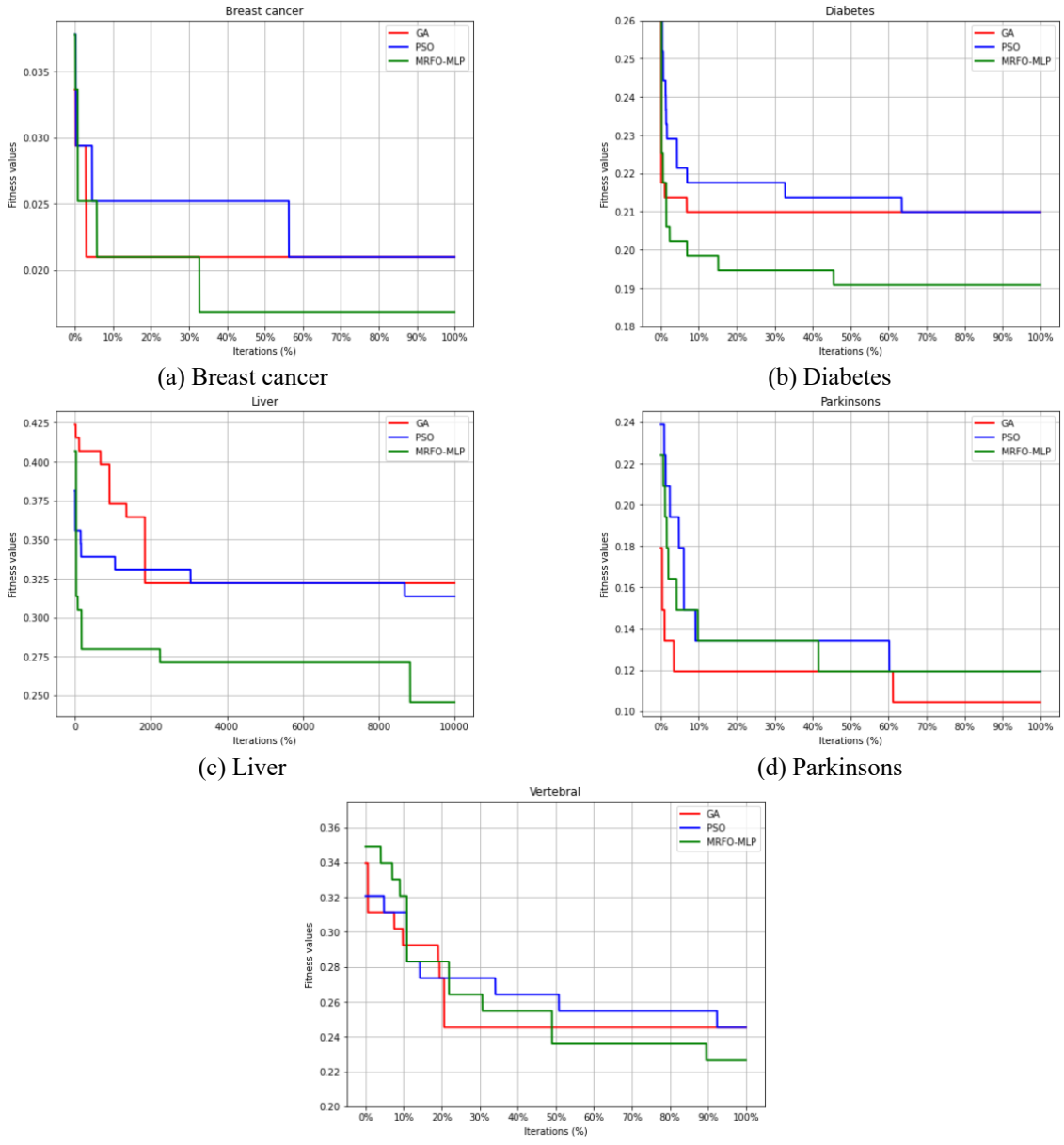


Figure 7. Relative performance improvement of MRFO-MLP

These findings demonstrate that incorporating the MRFO approach into MLP architecture optimisation yields a highly efficient classifier for medical data, surpassing some conventional ML

algorithms and other MHA methods, such as GA and PSO. The MRFO algorithm's strength in optimising solutions for complicated problems reflects an adequately tuned MLP model that can learn detailed patterns from the given dataset. Thus, the high average accuracy and low STD indicate that MRFO-MLP achieves high performance with lower variability, making it suitable for medical data classification. The relative performance improvement (in percentage) of the proposed MRFO-MLP model compared to GA-MLP and PSO-MLP across the five medical datasets is shown in Figure 7.

Two visualisations are provided to provide a complete analysis of the algorithm's behaviour. Figure 8 shows the trend of convergence of the best fitness value over the number of iterations, indicating the search speed and the capability of MRFO-MLP to overcome local optima compared to GA and PSO.



(a) Breast cancer

(b) Diabetes

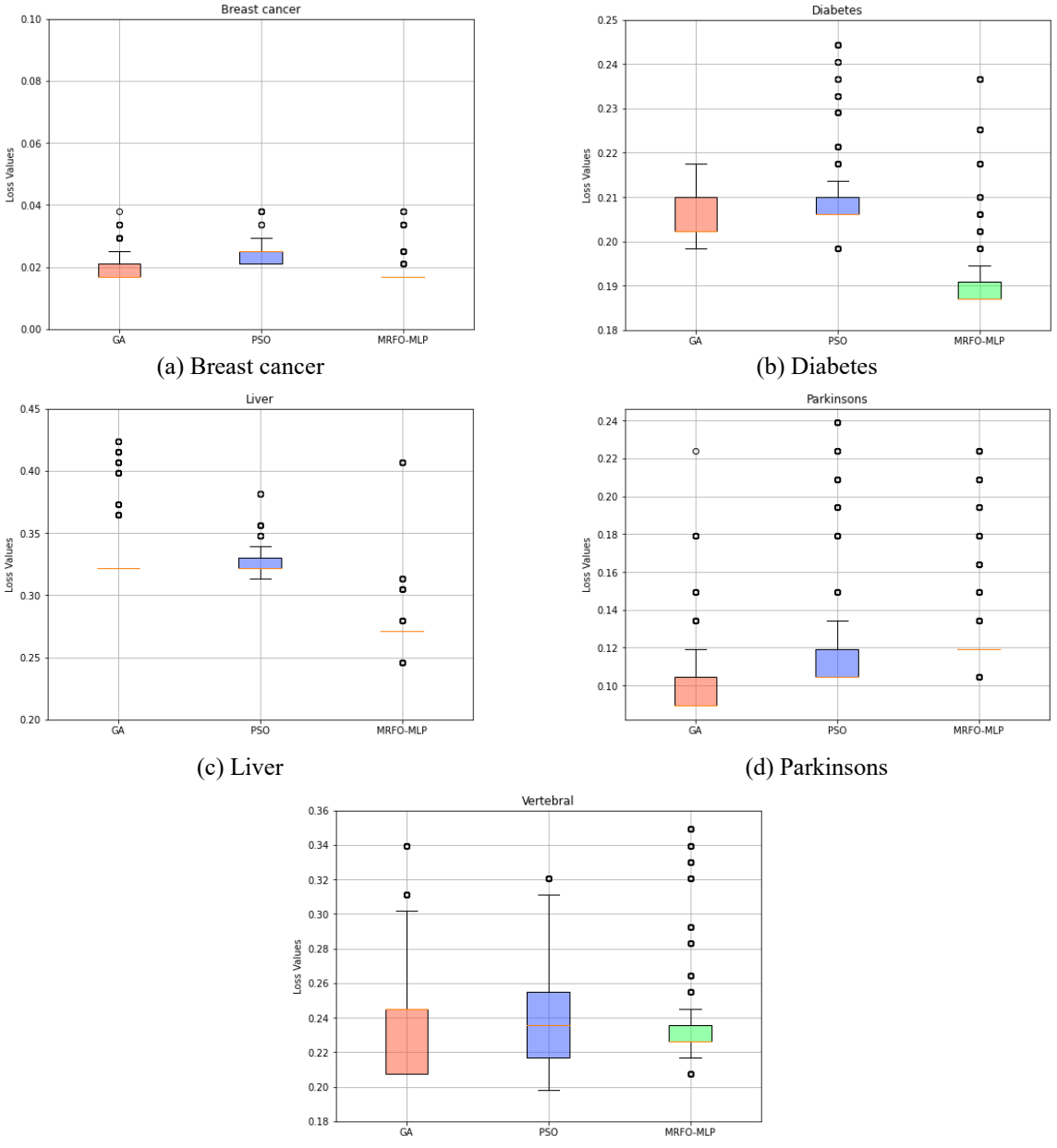
(c) Liver

(d) Parkinsons

(e) Vertebral

Figure 8. Convergence curves of fitness value for all datasets

Conversely, Figure 9 uses box plots to show the distribution of final fitness values across 30 independent runs. Whereas Figure 8 is based on the dynamic optimisation process, Figure 9 evaluates the robustness and stability of the model, which visually summarises the median, interquartile range (IQR), and outliers. Collectively, these statistics show that both MRFO-MLP converge faster and yield results with less variance.



(a) Breast cancer (b) Diabetes

(c) Liver (d) Parkinsons

(e) Vertebral

Figure 9. Box plots of fitness value for all datasets

To measure convergence beyond visual plots, we used Iteration Efficiency, the number of iterations to 95% of the final best accuracy. The smaller the value, the better the algorithm finds reasonable solutions with fewer functional evaluations.

Table 7 shows the average number of iterations required by each algorithm to reach this threshold across representative datasets. Based on the outcomes, MRFO-MLP is 2-6 times faster than the base approaches in terms of the number of generations. This efficacy has been credited to the interaction between the chain and cyclone foraging strategies, which balance global exploration and local exploitation more favourably than the usual operators in GA or PSO.

Table 7. Comparison of Iteration Efficiency (Avg. Iterations to reach 95% convergence threshold)

Dataset	Algorithm	Avg. Iterations	Speed-up Factor (vs. GA)
Breast Cancer	MRFO-MLP	18	6.4x
	PSO-MLP	42	2.7x
	GA-MLP	115	1.0x
Diabetes	MRFO-MLP	25	5.6x
	PSO-MLP	55	2.5x
	GA-MLP	140	1.0x
Parkinsons	MRFO-MLP	12	7.9x
	PSO-MLP	30	3.2x
	GA-MLP	95	1.0x
Vertebral	MRFO-MLP	21	6.0x
	PSO-MLP	48	2.6x
	GA-MLP	126	1.0x

Since the MHAs are stochastic, a two-tailed non-parametric Wilcoxon Signed-Rank Test [51] was performed to confirm the statistical significance of the performance improvements. The proposed MRFO-MLP was also evaluated against GA-MLP and PSO-MLP across 30 independent runs. Significance was assessed using a standard *p*-value threshold of $\alpha = 0.05$ for these independent pairwise comparisons.

The results are presented in Table 8 and include the p-value, test statistic (T), and outcome of the comparison (W). The W column represents the outcome: 1 indicates that MRFO-MLP is much better ($p < 0.05$), 0 indicates no significant difference ($p \geq 0.05$), and 2 indicates that the baseline is better. The last row (1/0/2) performs a summation of the total number of Wins, Ties, and Losses.

As shown in Table 7, the proposed MRFO-MLP shows substantial statistical superiority. Compared to GA-MLP, the model achieved statistical significance in 4 of 5 datasets (4/1/0), with no significant difference in Parkinson's ($p \geq 0.05$). Likewise, in the case of matching PSO-MLP, the proposed model won four datasets (4/1/0), the Liver dataset being the only loss (Tie). These results verify that the performance increase of the proposed framework is not due to random fluctuation, but it is very high and steady.

Table 8. Wilcoxon Signed-Rank test results of MRFO-MLP and the baseline algorithms

Dataset	MRFO-MLP vs. GA-MLP			MRFO-MLP vs. PSO-MLP		
	***p*-value**	***T***	***W***	***p*-value**	***T***	***W***
Breast Cancer	1.7257e-06	465	1	1.5224e-06	465	1
Diabetes	4.1669e-04	270	1	1.8040e-05	344	1
Liver	1.5609e-02	350	1	8.9682e-02	277.5	0
Parkinsons	2.8526e-06	210	0	1.3820e-04	36	1
Vertebral	1.8902e-04	414	1	1.6228e-04	368.5	1
1 / 0 / 2	**4 / 1 / 0**			**4 / 1 / 0**		

6. Conclusions

This study introduced MRFO-MLP, a novel model that uses the Manta Ray Foraging Optimisation (MRFO) algorithm to optimise the architecture of a Multi-layer Perceptron (MLP) neural network for medical data classification. The MRFO-MLP model uses a customised fitness function that effectively balances the trade-off between network performance and architectural complexity during optimisation, thereby preventing overfitting that may arise from overly complex network designs.

To evaluate the proposed model's performance, five medical datasets from the UCI repository were used. The classification accuracy of the MLP architecture generated using MRFO-MLP was compared with those of architectures optimised using the Genetic Algorithm (GA) and Particle Swarm Optimisation (PSO). The results demonstrate that the proposed MRFO-MLP model produces simpler architectures while achieving better or comparable performance across several evaluation aspects.

Furthermore, the MRFO-MLP model improves classification performance compared with traditional machine learning classifiers, including Naïve Bayes, Decision Trees, Random Forests, and Support Vector Machines. These findings indicate that MRFO-MLP is an effective approach for optimising MLP architectures in medical data classification tasks.

Future research may explore the application of MRFO-MLP for feature selection and hyperparameter tuning. In addition, the model could be extended to optimise MLP weights and biases, enabling it to function not only as an architecture optimisation tool but also as a comprehensive training strategy. Moreover, the proposed approach could be applied to other types of artificial neural networks, such as Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs), to optimise their architectural design further.

Data Availability Statement: The datasets analysed in this study are publicly available in the University of California, Irvine (UCI) Machine Learning Repository. Additional processed results are available from the corresponding author upon reasonable request.

Funding: This research was supported by Universiti Putra Malaysia (UPM) through its postgraduate research program.

Declaration of Competing Interest: The authors declare that they have no known competing interests.

Research Involving Human/Animal Participants: This article does not involve any studies conducted by the authors on animals or human participants.

References

[1] S. S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall, 1999.

[2] M. T. Alasaady, T. N. Mohd Aris, N. Mohd Sharef, and H. Hamdan, "A proposed approach for diabetes diagnosis using neuro-fuzzy technique," *Bulletin of Electrical Engineering and Informatics,* vol. 11, no. 6, pp. 3590-3597, 2022, doi: https://doi.org/10.11591/eei.v11i6.4269.

[3] P. Bansal, S. Gupta, S. Kumar, S. Sharma, and S. Sharma, "MLP-LOA: a metaheuristic approach to design an optimal multi-layer perceptron," *Soft Computing,* vol. 23, no. 23, pp. 12331-12345, 2019, doi: https://doi.org/10.1007/s00500-019-03773-2.

[4] H. T. Ünal and F. Başçiftçi, "Evolutionary design of neural network architectures: a review of three decades of research," *Artificial Intelligence Review,* vol. 55, no. 3, pp. 1723-1802, 2022, doi: https://doi.org/10.1007/s10462-021-10049-5.

[5] M. T. Al-Asaady, T. N. Mohd Aris, N. Mohd Sharef, and H. Hamdan, "XCOA-MLP: Extended Coyote Optimization Algorithm for Training Neural Networks in Medical Data Classification," *International Journal of Intelligent Engineering and Systems,* vol. 18, no. 11, pp. 746-762, 2025, doi: https://doi.org/10.22266/ijies2025.1231.46.

[6] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing,* vol. 415, pp. 295-316, 2020, doi: https://doi.org/10.1016/j.neucom.2020.07.061.

[7] R. Andonie, "Hyperparameter optimization in learning systems," *Journal of Membrane Computing,* vol. 1, no. 4, pp. 279-291, 2019, doi: https://doi.org/10.1007/s41965-019-00023-0.

[8] B. A. S. Emambocus, M. B. Jasser, and A. Amphawan, "A survey on the optimization of artificial neural networks using swarm intelligence algorithms," *IEEE Access,* vol. 11, pp. 1280-1294, 2023, doi: https://doi.org/10.1109/ACCESS.2022.3233596.

[9] M. T. Al-Asaady, T. N. Mohd Aris, N. Mohd Sharef, and H. Hamdan, "Recent advances on meta-heuristic algorithms for training multilayer perceptron neural network," *JOIV: International Journal on Informatics Visualization,* vol. 9, no. 2, pp. 658-673, 2025, doi: https://doi.org/10.62527/joiv.9.2.3109.

[10] F. Z. El-Hassani, Y. Ghanou, and K. Haddouch, "A novel model for optimizing multilayer perceptron neural network architecture based on genetic algorithm method," in *Artificial Intelligence and Industrial Applications*, vol. 772, (Lecture Notes in Networks and Systems. Cham, Switzerland: Springer, 2023, pp. 366-380, doi: https://doi.org/10.1007/978-3-031-43520-1_31

[11] M. N. Alam, A. Sallem, P. Pereira, B. Bachir, and N. Masmoudi, "Optimal artificial neural network using particle swarm optimization," 2023, doi: https://doi.org/10.1051/e3sconf/202346900019

[12] O. Altay and E. V. Altay, "A novel hybrid multi-layer perceptron neural network with improved grey wolf optimizer," *Neural Computing and Applications,* vol. 35, no. 1, pp. 529-556, 2023, doi: https://doi.org/10.1007/s00521-022-07775-4.

[13] F. Anka, F. S. Gharehchopogh, G. G. Tejani, and S. J. Mousavirad, "Advances in Mountain Gazelle Optimizer: A comprehensive study on its classification and applications," *International Journal of Computational Intelligence Systems,* vol. 18, no. 1, p. 247, 2025, doi: https://doi.org/10.1007/s44196-025-00968-4.

[14] F. Anka, N. Agaoglu, S. Nematzadeh, M. Torkamanian-afshar, and F. S. Gharehchopogh, "Advances in Artificial Rabbits Optimization: A comprehensive review," *Archives of Computational Methods in Engineering,* vol. 32, no. 4, pp. 2113-2148, 2025, doi: https://doi.org/10.1007/s11831-024-10202-7.

[15] M. T. Al-Asaady, T. N. Mohd Aris, N. Mohd Sharef, and H. Hamdan, "COA-MLP: Training neural networks using the Coyote Optimization Algorithm for medical data classification," presented at the 2025 7th International Conference on Interdisciplinary Computer Science and Engineering (ICICSE), Kuala Lumpur, Malaysia, 2025. [Online]. Available: https://doi.org/10.1109/ICICSE67247.2025.11390755

[16] F. Anka and N. Aghayev, "Advances in Sand Cat Swarm Optimization: A comprehensive study," *Archives of Computational Methods in Engineering,* vol. 32, no. 5, pp. 2669-2712, 2025, doi: https://doi.org/10.1007/s11831-024-10217-0.

[17] F. Anka, "A multi-strategy chimp optimization algorithm for solving global and constraint engineering problems," *Knowledge and Information Systems,* vol. 67, no. 8, pp. 6753-6802, 2025, doi: https://doi.org/10.1007/s10115-025-02422-5.

[18] M. F. Şahin and F. Anka, "Metaheuristics role in image processing and computer vision applications: a comprehensive review," *Cluster Computing,* vol. 28, no. 13, p. 871, 2025, doi: https://doi.org/10.1007/s10586-025-05610-8.

[19] F. Anka, "A novel hybrid metaheuristic method for efficient decentralised IoT network layouts," *Internet of Things,* vol. 32, p. 101612, 2025, doi: https://doi.org/10.1016/j.iot.2025.101612.

[20] F. Anka, G. G. Tejani, S. K. Sharma, and M. Baljon, "A bioinspired method for optimal task scheduling in fog-cloud environment," *Computer Modeling in Engineering & Sciences,* vol. 142, no. 3, pp. 2691-2724, 2025, doi: https://doi.org/10.32604/cmes.2025.061522.

[21] F. Kiani and Ö. F. Saraç, "A novel intelligent traffic recovery model for emergency vehicles based on context-aware reinforcement learning," *Information Sciences,* vol. 619, pp. 288-309, 2023, doi: https://doi.org/10.1016/j.ins.2022.11.057.

[22] L. Ma and K. Khorasani, "Constructive feed-forward neural networks using Hermite polynomial activation functions," *IEEE Transactions on Neural Networks,* vol. 16, no. 4, pp. 821-833, 2005, doi: https://doi.org/10.1109/TNN.2005.851786.

[23] M. Pelikan, *Hierarchical Bayesian Optimization Algorithm: Toward a New Generation of Evolutionary Algorithms*. Berlin, Germany: Springer, 2005, doi: https://doi.org/10.1007/b10910

[24] Z. B. Zabinsky, "Random search algorithms," in *Wiley Encyclopedia of Operations Research and Management Science*: John Wiley & Sons, 2011, doi: https://doi.org/10.1002/9780470400531.eorms0704

[25] N. Bacanin *et al.*, "Artificial neural networks hidden unit and weight connection optimization by quasi-refection-based learning artificial bee colony algorithm," *IEEE Access,* vol. 9, pp. 169135-169155, 2021, doi: https://doi.org/10.1109/ACCESS.2021.3135201.

[26] J. C. T. Arroyo and A. J. P. Delima, "An optimized neural network using genetic algorithm for cardiovascular disease prediction," *Journal of Advances in Information Technology,* vol. 13, no. 1, pp. 95-99, 2022, doi: https://doi.org/10.12720/jait.13.1.95-99.

[27] J. V. Domashova, S. S. Emtseva, V. S. Fail, and A. S. Gridin, "Selecting an optimal architecture of neural network using genetic algorithm," *Procedia Computer Science,* vol. 190, pp. 263-273, 2021, doi: https://doi.org/10.1016/j.procs.2021.06.036.

[28] A. A. M. Lima, F. K. H. de Barros, V. H. Yoshizumi, D. H. Spatti, and M. E. Dajer, "Optimized artificial neural network for biosignals classification using genetic algorithm," *Journal of Control, Automation and Electrical Systems,* vol. 30, no.

3, pp. 371-379, 2019, doi: https://doi.org/10.1007/s40313-019-00454-1.

[29] J. Cheng, J. Jiang, H. Kang, and L. Ma, "A hybrid neural architecture search algorithm optimized via lifespan particle swarm optimization for coal mine image recognition," *Mathematics,* vol. 13, no. 4, p. 631, 2025, doi: https://doi.org/10.3390/math13040631.

[30] D. Sarkar, T. Khan, and F. A. Talukdar, "Hyperparameters optimization of neural network using improved particle swarm optimization for modeling of electromagnetic inverse problems," *International Journal of Microwave and Wireless Technologies,* vol. 14, no. 10, pp. 1326-1337, 2022, doi: https://doi.org/10.1017/S1759078721001690.

[31] R. Jamous, H. AlRahhal, and M. El-Darieby, "Neural network architecture selection using Particle Swarm Optimization technique," *Applied Artificial Intelligence,* vol. 35, no. 15, pp. 1219-1236, 2021, doi: https://doi.org/10.1080/08839514.2021.1972251.

[32] X. Han, Y. Xue, Z. Wang, Y. Zhang, A. Muravev, and M. Gabbouj, "SaDENAS: A self-adaptive differential evolution algorithm for neural architecture search," *Swarm and Evolutionary Computation,* vol. 91, p. 101736, 2024, doi: https://doi.org/10.1016/j.swevo.2024.101736.

[33] S. Moradi, M. Gerdooei, S. M. Varedi-Koulaei, and H. G. Nosrati, "MLP neural network with an optimal architecture for modeling the ECAP-C process," *Neural Computing and Applications,* vol. 35, no. 3, pp. 2701-2715, 2023, doi: https://doi.org/10.1007/s00521-022-07685-5.

[34] O. I. Simsek and B. B. Alagoz, "Optimal architecture artificial neural network model design with exploitative alpha gray wolf optimisation for soft calibration of CO concentration measurements in electronic nose applications," *Transactions of the Institute of Measurement and Control,* vol. 45, no. 4, pp. 686-699, 2023, doi: https://doi.org/10.1177/01423312221119648.

[35] Y. Ghanou and G. Bencheikh, "Architecture optimization and training for the multilayer perceptron using ant system," *IAENG International Journal of Computer Science,* vol. 43, no. 1, pp. 10-18, 2016.

[36] M. Aviles, J. Rodríguez-Reséndiz, and D. Ibrahimi, "Optimising EMG classification through metaheuristic algorithms," *Technologies,* vol. 11, no. 4, p. 87, 2023, doi: https://doi.org/10.3390/technologies11040087.

[37] S. B. M. Sambatti, J. A. Anochi, E. F. P. da Luz, A. R. Carvalho, E. Shiguemori, and H. F. C. Velho, "MPCA meta-heuristics for automatic architecture optimisation of a supervised artificial neural network," presented at the 10th World Congress on Computational Mechanics, São Paulo, Brazil, 2012, doi: https://doi.org/10.5151/meceng-wccm2012-19075

[38] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation,* vol. 1, no. 1, pp. 67-82, 1997, doi: https://doi.org/10.1109/4235.585893.

[39] M. Rashid, O. M. Yaseen, R. R. Saeed, and M. T. Alasaady, "An improved ensemble machine learning approach for diabetes diagnosis," *Pertanika Journal of Science and Technology,* vol. 32, no. 3, 2024, doi: https://doi.org/10.47836/pjst.32.3.19.

[40] W. Zhao, Z. Zhang, and L. Wang, "Manta ray foraging optimisation: An effective bio-inspired optimiser for engineering applications," *Engineering Applications of Artificial Intelligence,* vol. 87, p. 103300, 2020, doi: https://doi.org/10.1016/j.engappai.2019.103300.

[41] F. Daqaq, R. Ellaia, M. Ouassaid, H. M. Zawbaa, and S. Kamel, "Enhanced chaotic manta ray foraging algorithm for function optimization and optimal wind farm layout problem," *IEEE Access,* vol. 10, pp. 78345-78369, 2022, doi: https://doi.org/10.1109/ACCESS.2022.3193233.

[42] S. Duman, A. Dalcalı, and H. Özbay, "Manta ray foraging optimisation algorithm-based feed-forward neural network for electric energy consumption forecasting," *International Transactions on Electrical Energy Systems,* vol. 31, no. 9, 2021, doi: https://doi.org/10.1002/2050-7038.12999.

[43] H. Faris, S. Mirjalili, and I. Aljarah, "Automatic selection of hidden neurons and weights in neural networks using grey wolf optimizer based on a hybrid encoding scheme," *International Journal of Machine Learning and Cybernetics,* vol. 10, no. 10, pp. 2901-2920, 2019, doi: https://doi.org/10.1007/s13042-018-00913-2.

[44] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, UK: Oxford University Press, 1995.

[45] V. K. Ojha, A. Abraham, and V. Snášel, "Metaheuristic design of feed-forward neural networks: A review of two decades of research," *Engineering Applications of Artificial Intelligence,* vol. 60, pp. 97-116, 2017, doi: https://doi.org/10.1016/j.engappai.2017.01.013.

[46] S. Mirjalili, "How effective is the Grey Wolf optimizer in training multi-layer perceptrons," *Applied Intelligence,* vol. 43, no. 1, pp. 150-161, 2015, doi: https://doi.org/10.1007/s10489-014-0645-7.

[47] M. Kelly, R. Longjohn, and K. Nottingham. "The UCI Machine Learning Repository", 2010, https://archive.ics.uci.edu (accessed Feb 12, 2025).

[48] M. T. Alasaady, M. G. Saeed, and K. H. Faraj, "Evaluation and comparison framework for data modeling languages," presented at the 2019 2nd International Conference on Electrical, Communication, Computer, Power and Control Engineering (ICECCPCE), 2019, doi: https://doi.org/10.1109/ICECCPCE46549.2019.203750

[49] M. G. Rojas, A. C. Olivera, and P. J. Vidal, "Optimising multi-layer perceptron weights and biases through a Cellular Genetic Algorithm for medical data classification," *Array,* vol. 14, p. 100173, 2022, doi: https://doi.org/10.1016/j.array.2022.100173.

[50] G. H. de Rosa, D. Rodrigues, and J. P. Papa, "Opytimizer: A nature-inspired Python optimizer," vol. arXiv:1912.13002*,* doi: https://doi.org/10.48550/arXiv.1912.13002.

[51] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of non-parametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation,* vol. 1, no. 1, pp. 3-18, 2011, doi: https://doi.org/10.1016/j.swevo.2011.02.002.