

Swarm-Based Extreme Learning Machine Models for Global Optimization

Mustafa Abdul Salam^{1,*}, Ahmad Taher Azar² and Rana Hussien²

¹Artificial Intelligence Department, Faculty of Computers and Artificial Intelligence, Benha University, Benha, 13518, Egypt

²Scientific Computing Department, Faculty of Computers and Artificial Intelligence, Benha University, Benha, 13518, Egypt

*Corresponding Author: Mustafa Abdul Salam. Email: mustafa.abdo@fci.bu.edu.eg

Received: 30 May 2021; Accepted: 29 July 2021

Abstract: Extreme Learning Machine (ELM) is popular in batch learning, sequential learning, and progressive learning, due to its speed, easy integration, and generalization ability. While, Traditional ELM cannot train massive data rapidly and efficiently due to its memory residence, high time and space complexity. In ELM, the hidden layer typically necessitates a huge number of nodes. Furthermore, there is no certainty that the arrangement of weights and biases within the hidden layer is optimal. To solve this problem, the traditional ELM has been hybridized with swarm intelligence optimization techniques. This paper displays five proposed hybrid Algorithms “Salp Swarm Algorithm (SSA-ELM), Grasshopper Algorithm (GOA-ELM), Grey Wolf Algorithm (GWO-ELM), Whale optimization Algorithm (WOA-ELM) and Moth Flame Optimization (MFO-ELM)”. These five optimizers are hybridized with standard ELM methodology for resolving the tumor type classification using gene expression data. The proposed models applied to the predication of electricity loading data, that describes the energy use of a single residence over a four-year period. In the hidden layer, Swarm algorithms are used to pick a smaller number of nodes to speed up the execution of ELM. The best weights and preferences were calculated by these algorithms for the hidden layer. Experimental results demonstrated that the proposed MFO-ELM achieved 98.13% accuracy and this is the highest model in accuracy in tumor type classification gene expression data. While in predication, the proposed GOA-ELM achieved 0.397 which is least RMSE compared to the other models.

Keywords: Extreme learning machine; salp swarm optimization algorithm; grasshopper optimization algorithm; grey wolf optimization algorithm; moth flame optimization algorithm; bio-inspired optimization; classification model; and whale optimization algorithm

1 Introduction

In recent years, Extreme Learning machine (ELM) has attracted a significant attention in a range of fields as an accurate and powerful machine learning technique. ELM is a less-square learning algorithm for a ‘general’ single-hidden layer feedforward network (SLFN) that can be



This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

used as an estimator or as a classifier in a regression problem. ELM has two major advantages: easier to understand and excellent results in generalization. ELM has also been used to carry out many studies in the fields of image applications filter design, market predictions, time series forecasting, energy efficiency shipping, electricity load prediction, target detection, aircraft reconnaissance, clustering, real-time error diagnostics, end-point prediction model, design of neural architectures, illness diagnosis, agility prevision, etc [1].

The Single Hidden Layer Neural Feed Forward (SLFN) is commonly considered as one of the most popular learning machine models in the classification and prediction areas. The Learning Algorithm is widely acknowledged to be at the core of the Neural Network [2]. Traditional gradient-based machine learning approaches, such as Levenberg-Marquardt (LM) and Scaled-Conjugate-Gradient (SCG), complain about over-fitting, minima local, and long-term waste memory [3]. ELM [4] has been presented to fix complex issues mentioned in gradient-based machine learning algorithms. ELM is used as an SLFN learning solution. Although many real-life problems have been arisen, ELM has enormous precision and anticipation speed [5]. Instead of turning all inner parameters as in gradient-based algorithms, ELM efficiently determines the weight of inputs and the bias of the hidden layer. ELM also assists in offering an unbiased overview of the output weight. Gradient-based learning algorithms need less hidden neurons than ELM due to the erratic choice of input weights and hidden layer bias [6,7].

1.1 Motivations and Contributions

Machine learning (ML) has proven to be useful in tackling tough issues in classification and prediction. ELM, one of the machine learning algorithms that requires significantly shorter training time than popular Back Propagation (BP) and support vector machine/support vector regression (SVM/SVR) [8]. In many situations, the prediction accuracy of ELM is somewhat greater than that of BP and close to that of SVM/SVR. In comparison to BP and SVR, ELM is easier to construct since there is few parameters to tune. ELM requires more hidden nodes than BP but many fewer than SVM/SVR, implying that ELM and BP respond to unknown input considerably faster than SVM/SVR. ELM has an incredibly rapid learning rate. In contrast to standard gradient-based learning methods, which only operate with differentiable activation functions. Unlike standard gradient-based learning algorithms, which have numerous difficulties such as local minima, inappropriate learning rate, and overfitting, etc., ELM tends to achieve simple answers without such minor issues. Many learning algorithms, such as neural networks and support vector machines, appear considerably easier than the ELM learning method. The ELM's obstacles overcame, by using bio-inspired algorithms to refine it.

Five bio-inspired algorithms are proposed to optimize ELM model. These algorithms are namely, Salp Swarm Optimizer (SSA) [9], Grasshopper Optimizer (GOA) [10], Grey Wolf Optimizer (GWO) [11], Moth Flame Optimizer (MFO) [12] and Whale Optimizer (WOA) [13]. These algorithms have been used to select input weights and biases in order to define ELM weights of the output. Proposed models performed a significant generalization of the simplified structure. They have been used to identify the quality of motions of patients with distinctive tumor forms and their advanced identification. Also, they have been used to predict the electricity consumption of Clients.

1.2 Paper Organization

The paper is structured as follows: Section 2 devoted for related work. Section 3 presents preliminaries that describe (SVM, ELM and the swarm intelligence algorithms); Section 5 reflects

on the potential approach and the implementation of hybrid models; Section 6 discusses the experimental results, while Section 7 points out the core conclusions of the proposed model and highlights the future work.

2 Related Work

Wang et al. [8] investigated the efficacy of the ELM and suggested a revised effectiveness of extreme learning algorithm. The suggested algorithm requires an optimal set of input weights and assumptions, ideally maintaining the full column rank of H matrix. Isham et al. [14] used a whale optimization algorithm (WOA) technique to determine the optimal value of the input weight and the hidden layer distortions. Huang et al. [15] proposed a new learning algorithm called the ELM for single-hidden layers feedforward neural networks (SLFNs), which randomly chooses hidden nodes and analytically measures SLFN output weights. This algorithm generalized at an extremely high pace in theory while studying. Using ELM ideas. HuaLing et al. [16] increased the performance of ELM convergence by integrating ELM and PSO algorithm. Alhamdoosh et al. [17] presented an evolutionary approach that was defined for the assembly of ELM to reach an optimal solution while preserving the size of the ensemble under control. Wu et al. [18] used GA algorithm to optimize ELM model. Sundararajanc et al. [19] used a truly modern genetic algorithm called ‘RCGA ELM’ to pick out the strongest neurons in the hidden layer. Zhao et al. [20] suggested a genetic ELM focused on the economic distribution of the power grid. Abdul Salam et al. [21] refined the ELM model and increased efficiency in stock market forecasts compared to the classic ELM model, as well as they suggested a new Flower Pollination Algorithm (FPA). Salam et al. [22] stated that in this model, the dragonfly optimizer applied a hybrid dragonfly algorithm with an extreme predictive learning scheme, which improved the ELM model’s predictive performance. According to Aljarah et al. [23], the proposed WOA-based training algorithm outperforms existing algorithms in most datasets; not only in terms of accuracy, but also in terms of convergence. Mohapatra et al. [24] discovered that ELM-based classifiers work better when projecting higher dimensional space elements. Liu et al. [25] proposed a multi-extension of the kernel ELM to manage heterogeneous databases through kernel tuning and convergence. According to Huang et al. [26], the separating hyperplane begins to travel through the root of the ELM’s space function, resulting in less space. SVM has less optimization limitations and improved generalization performance. It was also noted that there is a generalization of the ELM’s output that is less sensitive to learning parameters such as the number of hidden nodes. Zong et al. [27] demonstrated that the weighted ELM is capable of balanced data generalization. Sowmya et al. [28] designed sensor system uses the spectroscopic method using machine learning to improve the accuracy of detection.

3 Preliminaries

3.1 Extreme Learning Machine (ELM)

The Extreme learning machine for training SLFN was proposed by Huang et al. [15]. ELM’s central idea is the hidden weights of the layer. In addition, the biases are generated randomly, with the least-squares procedure used to determine the output weights. The outputs of the objectives and the hidden layer [29] are also established. Fig. 1 shows a high-level outline of the training algorithm and the ELM layout. Where: N refers to a set of unique samples (X_i, t_i) , where $X_i = [x_{i1}, x_{i2} \dots x_{in}]^T \in R^n$ and $t_i = [t_{i1}, t_{i2} \dots t_{im}]^T \in R^m$, L represents the hidden layer nodes, $g(x)$ represents the activation function,

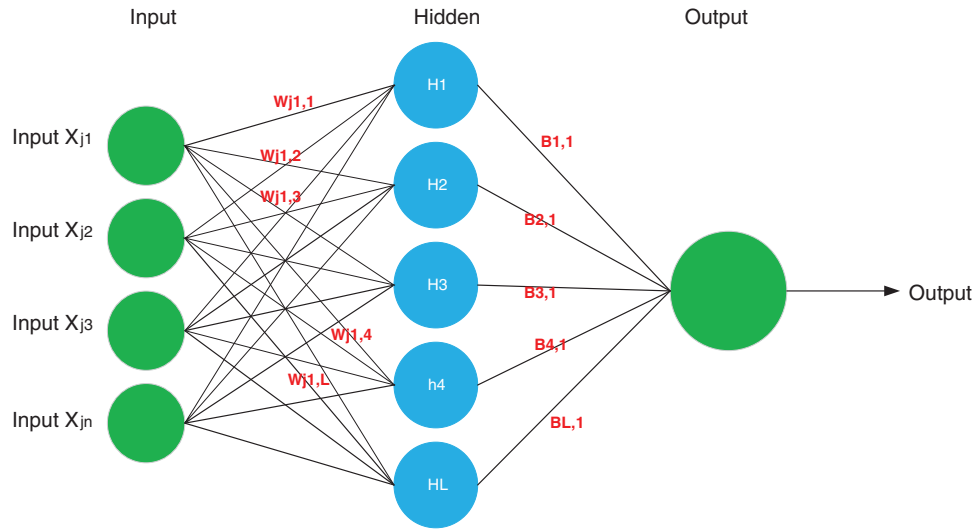


Figure 1: Diagram of ELM (Huang et al. 2011)

There is also a mathematical model represented as shown in Eq. (1):

$$\sum_{i=1}^L \beta_i g_i(X_j) = \sum_{i=1}^L \beta_i g_i(W_i \cdot X_j + b_i) \quad (1)$$

where: $W_i = [W_{i1}, W_{i2} \dots W_{in}]^T$ represents the vector of weight connecting the hidden node and the i th nodes, $\beta_i = [\beta_{i1}, \beta_{i2} \dots \beta_{im}]^T$ represents the vector weight which connects the hidden nodes to the output nodes, b_i It represents the threshold of the hidden nodes, $W_i \cdot X_j$ represents the inner product of W_i and X_j . Selection of the output nodes is linearly done, however.

The SLFN and L hidden nodes norm in the $g(x)$ activation function is error-free. In other words, mean: $\sum_{j=1}^L \|o_j - t_j\| = 0$, i.e., and there exist β_i , W_i and b_i in such a way that declared in Eq. (2)

$$\sum_{i=1}^L \beta_i g_i(W_i X_j + b_j) = t_j, \quad j = 1, \dots, N, \quad (2)$$

It can then be presented as represented in Eq. (3)

$$H\beta = T \quad (3)$$

where we can represent H Matrix as represented below in Eq. (4)

$$H(W_1 \dots W_L, b_1 \dots b_L, X_1 \dots X_N) = \begin{bmatrix} g(W_{.1} \cdot X_1 + b_1) \cdots g(W_{.L} \cdot X_1 + b_L) \\ \vdots \quad \quad \quad \vdots \quad \quad \quad \vdots \\ g(W_{.1} \cdot X_N + b_1) \cdots g(W_{.L} \cdot X_N + b_L) \end{bmatrix} \quad (4)$$

We can display β in Eq. (5) and T in Eq. (6)

$$\beta = \begin{bmatrix} \beta_1^T \\ \beta_L^T \end{bmatrix}_{L \times m} \quad (5)$$

$$T = \begin{bmatrix} t_1^T \\ \vdots \\ t_{LN}^T \end{bmatrix}_{N \times m} \quad (6)$$

And Eq. (3) becomes a linear system. Furthermore, by finding a least square solution, the performance weights β can be calculated analytically: $\beta = H^\dagger T$ Where H^\dagger Is the generalized inverse for H Moore–Penrose.

This results in a mathematical transformation in the weighing of the items. This ensures that a change in network parameters for such basic learning parameters eliminates the long training time (such as iterations and learning rate).

A number of variables were described, Huang et al. (2006), The H matrix in which the hidden layers are represented and the i th column determines the hidden layer input nodes. The activation function g becomes infinitely differentiable if $L \leq N$ represents the optimal number of hidden nodes.

3.2 Swarm Intelligence Algorithms

3.2.1 Salp Swarm Algorithm (SSA)

Salp Swarm Algorithm (SSA) was proposed by Mirjalili et al. [9]. The main inspiration of SSA is the swarming behaviour of salps when navigating and foraging in oceans. The society is divided into two groups: leaders and followers to model the salp chains mathematically. The salp is the leader in the front chain, while the remaining salp followers are called.

The leader's position can be updated according to Eq. (7) as follows:

$$x_j^1 = \begin{cases} F_j + c_1((ub_j - lb_j)c_2 + lb_j) \\ F_j - c_1((ub_j - lb_j)c_2 + lb_j) \end{cases} \quad (7)$$

where x_j^1 represents the position of the first salp (leader) in the j th dimension, F_j is the position of the food source in the j th dimension, ub_j represents the upper bound of j th dimension, lb_j represents the lower bound of j th dimension, c_1 , c_2 , and c_3 are random numbers.

Eq. (8): Newton's law of motion was used to update the position of the followers:

$$x_j^i = \frac{1}{2}at^2 + v_0t \quad (8)$$

where $i \geq 2$, x_j^i shows the position of i th follower salp in j th dimension, t is time, v_0 is the initial speed, and $a = \frac{v_{final}}{v_0}$ where $v = \frac{x-x_0}{t}$.

3.2.2 Grasshopper Optimization Algorithm (GOA)

Luo et al. [10] created the GOA, a new swarm intelligence algorithm. GOA is a system that is dependent on the population. GOA mimics the nature and social activity of grasshopper swarms.

Three forces determine each grasshopper's place in the swarm. Its social contact with the other grasshoppers S_i , Gravitational effect on it G_i , and wind advection the letter A_i . The following Eq. (9) is the final shape of the three impacted powers on each grasshopper.

$$X_i = S_i + G_i + A_i \quad (9)$$

Eq. (10) explains the social interaction between each grasshopper and the other grasshopper.

$$S_i = \sum_{j=1}^N s(d_{ij}) \hat{d}_{ij}, \quad j \neq i \quad (10)$$

where d_{ij} is the distance between the grasshopper $_i$ and grasshopper $_j$, s is a function represents the strength of two social force, attraction and repulsion between grasshoppers, it can be defined as represented in Eq. (11).

$$s(r) = f e^{\frac{-r}{l}} - e^{-r} \quad (11)$$

where f , l is the attraction strength and the attractive length dimension. The position of the grasshopper can be calculated as declared in Eq. (12):

$$X_i = \sum_{j=1}^N s(|x_j - x_i|) \frac{x_j - x_i}{d_{ij}} - g \hat{e}_g + u \hat{e}_w \quad (12)$$

3.2.3 Whale Optimization Algorithm (WOA)

WOA could be a recent algorithm introduced in 2016 by Mirjalili et al. [13] on a population level. WOA is derived from the social activity of humpback whales.

The difference is determined between the whale at (X, Y) and the abuser at (X^*, Y^*) . At this point, after the Helix-shaped forming of the Humpback whale, a winding condition is formed between the position of the whale and the victim to be taken as described below in Eqs. (13) and (14):

$$\vec{X}(t+1) = e^{bk} \cdot \cos(2\pi k) \cdot \vec{D}^* + \vec{X}^*(t), \quad (13)$$

$$\vec{D}^* = |\vec{X}^*(t) - \vec{X}(t)| \quad (14)$$

To retrieve the pattern of the logarithmic spiral, if B can be a constant value, and k can be an arbitrary number within the range $[-1, 1]$. This way, in the middle of optimization, WOA thinks about changing whaling. There is a 50 percent chance of choosing between the lease component and the contract portion, and the contract components are as represented in Eq. (15):

$$\vec{X}(t+1) = \begin{cases} \vec{X}^* - \vec{A} \cdot \vec{D} \\ e^{bk} \cdot \cos(2\pi k) \cdot \vec{D}^* + \vec{X}^*(t), \quad \text{if } p \geq 0.5, \end{cases} \quad (15)$$

where p is a random number within the $(0,1)$.

3.2.4 Grey Wolf Optimizer (GWO)

Seyedali Mirjalili and colleagues proposed the grey wolf optimizer (GWO) in 2014 as a novel swarm intelligent optimization algorithm that primarily imitates the hierarchy of wolf management and hunting in nature [30].

Mirjalili developed the optimization algorithm by modelling it after the looking and hunting actions of grey wolves. The optimum solution is defined as alpha (α), the second best as beta

(β), and the third best as delta (δ) for the mathematical model. Many of the remaining candidate solutions are omegas (ω). These three grey wolves will lead all the omegas when looking (optimizing) and shooting.

When a prey object is discovered, the iteration starts (t = 1). Following that, the alpha, beta, and delta wolves would lead the omegas in pursuing and ultimately encircling the prey. To define the encircling action, three coefficients are proposed as represent in Eqs. (16)–(18):

$$\vec{D}_\alpha = |\vec{C}_1 \cdot \vec{X}_\alpha - \vec{X}(t)|, \tag{16}$$

$$\vec{D}_\beta = |\vec{C}_2 \cdot \vec{X}_\beta - \vec{X}(t)|, \tag{17}$$

$$\vec{D}_\delta = |\vec{C}_3 \cdot \vec{X}_\delta - \vec{X}(t)|, \tag{18}$$

where t indicates the current iteration, \vec{X} is the position vector of the grey wolf, and \vec{X}_1 , \vec{X}_2 & \vec{X}_3 are the position vectors of the alpha, beta, and delta wolves. \vec{X} would be computed as Eqs. (19)–(22):

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \cdot \vec{D}_\alpha, \tag{19}$$

$$\vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \cdot \vec{D}_\beta \tag{20}$$

$$\vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \cdot \vec{D}_\delta \tag{21}$$

$$\vec{X}(t) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{22}$$

3.2.5 Moth–Flame Optimization Algorithm (MFO)

MFO is a new Metaheuristic population-based approach created by Mirjalili in 2015 [31] that mimics the transverse orientation for navigation technique used by moths at night. Moths travel at night dependent on the moonlight, keeping a set angle to navigate their way.

MFO starts by randomly creating moths in the solution space, then calculating each moth’s fitness values (i.e., location) and labelling the best position with flame. The position of the moths is then updated with the spiral movement function to attain better flame marked locations, then the best individual location updates and the previous processes continue until the ends have been achieved. MFO converges the global optimum of optimization problems using three associated functions. The following are these functions as declared in Eq. (23):

$$MFO = (I, P, T) \tag{23}$$

where I refers to the first random locations of the moths (I: $\Phi \rightarrow M$, OM), P refers to motion of the moths in the search space (P: $M \rightarrow M$), and T refers to finish the search process (T: $M \rightarrow$ true, false). Eq. (24) represents I function, which is used for implementing the random distribution.

$$M(i, j) = (ub(i) - lb(j)) * rand() + lb(i) \tag{24}$$

where lb and ub indicate the lower and upper bounds of variables, respectively.

The MFO algorithm's logarithmic spiral is as described in Eq. (25):

$$S(M_i, F_j) = D_i \cdot e^{bt} \cdot \cos(2\pi t) + F_j \quad (25)$$

where D_i refers to the space between the i -th moth and the j -th flame (i.e., $D_i = |F_i - M_i|$), b indicates a fix to define the shape of the logarithmic spiral, and t indicates a random number between $[-1, 1]$.

This section focuses on enhancing MFO algorithm exploitation (for example, updating the moths' positions in n different locations in the search space will minimize the probability of exploiting the most promising solutions). Reducing the number of fires, as a consequence of the following Eq. (26), helps in the resolution of this problem:

$$flame\ no = round\left(N - l * \frac{N - l}{T}\right) \quad (26)$$

where N denotes the maximum number of fires, l the present number of iterations, and T the cumulative number of iterations.

4 The Proposed Hybrid Models

As long as a suitable number of hidden nodes are selected within the program, ELMs offer the benefit of taking minimum training time while still preserving acceptable classification and predication. Since the optimum hidden layer weighting is not achievable within the hidden layer, the huge number of nodes slows down the execution of the ELM experiments. Therefore, in this hybrid model, fewer nodes within the hidden layer must be used to accelerate ELM execution while retaining an appropriate choice of hidden layer weights and predispositions. It is often used in the same manner when determining the weights and preferences of the output sheet. Swarm algorithms are the most efficient for collecting the hidden layer weights and expectations that optimize the overall execution of the ELM.

4.1 SSA-ELM Model

The hidden output layer matrix' pseudo reverse matrix serves to establish the weight of the output layer matrix of an ELM and over-fit a significant amount of hidden layer nodes. In addition, the weight and hidden offset layer value of the ELM are randomly generated, resulting in multiple weights matrices of the output layer. The Salp Swarm Algorithm optimizes input weight and hidden layer bias values in the ELM to avoid confusion created by random selection of the two. Hence, iterative update and optimization [32] offered the globally best approach.

According to the literature [33], this section addresses the nature and implementation of the SSA - ELM training Algorithm. The SSA is used for the optimization of the ELM network, which serves the ELM candidate network for any salp path. SSA is programmed to maintain the parameters of the network to be optimized, namely the weights of a connection between the input and the hidden level and the bias of the neurons in the hidden layer. As a result, the length of each salp can be determined with the formula $L = I * N + N$, where I denotes the number of input variables. Fig. 2 depicts the architecture of the SSA-structural ELM.

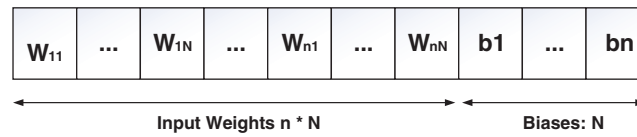


Figure 2: Structural design of SSA-ELM

Algorithm 1: Hybrid Salp Swarm Optimization and Extreme Learning Machine (SSA-ELM)

Inputs:

- N: the number of Agents
- T: Iterations Number
- X^*Z : the best search agent
- P: The current search position
- Trn.: Training Data set
- Vld.: Validation Data set

Outputs:

- f_{best} : Optimal hidden weights and biases
- $f(f_{best})$: sum square error for the NN over the validation set f_{best}

Initialize:

- Generate X_i Initial Population ($i = 1, 2, \dots, n$) considering ub and lb.
- Calculate each solution's fitness.
- X^* = the ideal agent for a search

while (The final criterion is not fulfilled)

Calculate each search agent's fitness (Salp)

F= Best Agent for Search

Update c1

for each salp (xi)

if ($i = 1$)

Modify the lead salp position.

Else

Modify the follower's position.

End

End

Modify Salps depending on the higher and lower limits.

End

Return best ELM network with minimum RMSE.

Return F

4.2 GOA-ELM Model

The hybrid ELM model based on GOA is mentioned in detail in this section. GOA is used in GOA-ELM to refine the parameters of ELM. The suggested technique is split into two sections. The first stage entails refining internal parameters. The second segment assesses the efficiency of the outer classification. The GOA technique dynamically adjusts the optimum parameters for the training set while tuning the inner parameters for ELM. The optimal parameters are then fed into the ELM model in the outer loop to perform the classification task.

Algorithm 2: Hybrid Grasshopper Optimization and Extreme Learning Machine (GOA -ELM)

Inputs:

- N: the maximum number of Agents
- T: Number of Iterations
- X^* : the best search agent
- P: The current search position
- T rn.: Training Data set
- V Id.: Validation Data set

Outputs:

- f_{best} : Optimal hidden weights and biases
- $f(f_{best})$: sum square error for the NN over the validation set f_{best}

Initialize:

- Generate X_i Initial Population ($i = 1, 2, \dots, n$)
- Initialize c_{max} , c_{min} and L (maximum number of iterations);
- Evaluate the fitness of all agents by ELM with agent as parameters;
- Calculate each solution's fitness
- TZ the ideal agent for a search

while ($l \leq L$)

Update c

for $i < n$

Standardize the grasshopping distance.

Update the search agent's position X_i

Perform Levy flight strategy.

Bring the X_i back if it goes outside the boundaries.

end for

Assess the fitness of the agent as parameters of all agents with ELM

If a better option is available, update T.

for $i < n$

Calculate the oppositional position x_i^{op} of X_i

Assess for this agent as criteria the fitness for the existing ELM search agent.

Return the new search agent as it exceeds borders.

end for.

Update T if there is a better solution.

Update population X by selecting the best n search agents from X^{op} and X

$l = l + 1$

end while

Return T as ELM's best parameter.

4.3 GWO-ELM Model

This algorithm can be paired with other optimization algorithms to solve these issues and increase the performance of ELM. The ELM algorithm will be paired with GWO in this analysis

since it has a sufficient convergence rate and does not have several tuning parameters. ELM, on the other hand, can be paired with other optimization algorithms. The following steps can be taken to build a hybrid ELM-GWO algorithm:

Algorithm 3: Hybrid Grey Wolf Optimization and Extreme Learning Machine (GWO -ELM)

Inputs:

- N: the maximum number of Agents
- T: Number of Iterations
- X^* : the best search agent
- P: The current search position
- Trn.: Training Data set
- Vld.: Validation Data set

Outputs:

- f_{best} : Optimal hidden weights and biases
- $f(f_{best})$: sum square error for the NN over the validation set f_{best}

Initialize:

- Generate X_i Initial Population ($i = 1, 2, \dots, n$)
- Calculate each solution's fitness
- X^* = the ideal agent for a search

Record the best individual, second best individual and third best individual as $\vec{X}_\alpha, \vec{X}_\beta$ & \vec{X}_δ

While ($t < \text{maximum iteration number}$)

For each individual

Update the position of current individual by Eqs. (D.1)–(D.7)

End for

Update a, A, C

Calculate the fitness value of all individuals in the population

Update $\vec{X}_\alpha, \vec{X}_\beta$ & \vec{X}_δ

$t = t + 1$

End While

Return \vec{X}_α

4.4 MFO-ELM Model

The proposed MFO-ELM model retains the same method for determining the output sheet's weights and preferences. The MFO algorithm is used to maximize the overall efficiency of the ELM by choosing the weights and expectations of the secret layer as described in Eq. (27). M variables must be calculated by the MFO, with $M = (N_H + 1) * N_I$ where N_H and N_I are the total number of nodes in the secret and input layers, in that order. Individual vector ranges are limited to $[-1, 1]$.

$$F(w) = \sum_{j=1}^{N_{Validation}} \sum_{i=1}^{N_o} |O_{ij} - d_{ij}| \quad (27)$$

where $N_{Validation}$ is the number of data points in the validation set, N_o is the number of nodes in the output layer. O_{ij} is the actual output on node i as a result of applying sample j to the ELM, and d_{ij} is the desired output on node i for a sample j .

Algorithm 4 presents an algorithm that depicts the proposed MFO-ELM model as follows:

Algorithm 4: Hybrid Moth Flame Optimization and Extreme Learning Machine (MFO-ELM)

Inputs:

- n: Number of moths (search agents)
- N The maximum number of flames
- T: Number of Iterations
- X^* : the best search agent
- T rn.: Training Data set
- V ld.: Validation Data set

Outputs:

- f best: Optimal hidden weights and biases
- f (f best): sum square error for the NN over the validation set f best

Initialize:

- initialize population of n flames positions randomly with hidden layers weight and biases.
- Initialize the parameters for Moth-flame.

while Criteria not met **do**

Update the No. of flames to be used N_{flames}
according to equation E.6.

Fitness of all n moths is determined by

foreach $Moth_i$ with $i \leq n$ **do**

- create NN with the weights and biases of the hidden layer of $Moth_i$

- Calculate weights of output layer with the given training data from the MP matrix Trn
and hidden layer weights and biases.

- Asses NN model using the validation data V ld

End

If first iteration, **then**

Sort the moths as fit and put the product into the flame matrix.

Else

- Combine the population of pasr moths and flames.

- The combined population is ranked between best and worst.

- Choose from the classified combined population the best N places as flames.

End

Calculate the convergence constant r

foreach $Moth_i$ with $i \leq n$ **do**

Calculate t as $t = (r-1) * rand + 1$

If $i \leq N$ **then**

Update $Moth_i$ Position according to $Flame_i$ using Eq. E.5

Else

Update $Moth_i$ Position according to $Flame_{N_{flames}}$ using Eq. E.5

End

End

End

4.5 WOA-ELM Model

The suggested WOA-ELM algorithm consists of the following steps:

1- Initialize all the humpback whales ($z = 1, \dots, P_s$) randomly where each whale represents a set of input weights and hidden biases within a range $[-1, 1]$ as represented in Eq. (28)

$$s_z = [w_{11}^h, \dots, w_{1l}^h, w_{n_h1}^h, \dots, w_{n_hl}^h, b_1, b_2, \dots, b_{nh}] \quad (28)$$

2- Evaluate the output weights and fitness of each whale and find s_{best} of all the solutions in the population. The fitness is defined as represented in Eq. (29)

$$f(s_{best}) = \sqrt{\frac{\sum_{j=1}^{N_v} \|\sum_{i=1}^{n_h} w_i^o \varnothing(w_i^h \cdot x_j + b_i) - t_j\|_2^2}{N_v}} \quad (29)$$

where, N_v indicates the number of validation samples

3- For each whale, calculate a , A , ζ , q , and pr . Update the position based on Eq. (24) and generate the new population.

4- Bound the new solutions using the following Eq. (30).

$$s_z(k+1) = \begin{cases} -1, & s_z(k+1) < -1, \\ +1, & s_z(k+1) > +1 \end{cases} \quad (30)$$

and find the best new solution $s_{bestnew}$.

5- Update the s_{best} using the norm of the output weights and the fitness value as (31).

$$s_{best} = \begin{cases} s_{bestnew}, & \text{if } (|f(s_{best}) - f(s_{bestnew})| < \epsilon f(s_{best}) \text{ and } w_{s_{bestnew}}^o) \\ s_{best}, & \text{otherwise} \end{cases} \quad (31)$$

where $f(s_{best})$ and $f(s_{bestnew})$ denotes the fitness of the best solution so far and the new best solution respectively. $w_{s_{best}}^o$ and $w_{s_{bestnew}}^o$ represents the output weights of best solution so far and the best new solution respectively. $\epsilon > 0$ is a user-defined tolerance rate.

6- Repeat steps 3–5 before the end condition is not met (i.e., the full number of iterations) to achieve the desired input weights and latent biases.

5 Experimental Results

5.1 Dataset Description

With the classification and predication information package of the UCI Machine Learning store used in the experiments and comparisons. Several highlights and incidents were chosen to provide the set of information as agents with various kinds of issues to be addressed in the strategy illustrated. To ensure that optimization algorithms are applied in colossal search spaces as seen in Tabs. 1 and 2, we picked a set of high-dimensional results separately.

We worked on two datasets, one for classification and the other for prediction.

Algorithm 5: Hybrid Whale Optimization and Extreme Learning Machine (WOA-ELM)

Inputs:

- N: the maximum number of Agents
- T: Number of Iterations
- X^* : the best search agent
- P: The current search position
- Trn.: Training Data set
- Vld.: Validation Data set

Outputs:

- f_{best} : Optimal hidden weights and biases
- $f(f_{best})$: sum square error for the NN over the
- validation set f_{best}

Initialize:

- prepare the training data and testing data.
- set the relevant parameters of ELM.
- Initialize the WOA population
- Compute the fitness of each search agent.
- X^* = the ideal agent for a search

While ($t < \text{Max_Iteration}$) **for** each solution Modify \mathbf{a} , \mathbf{A} , \mathbf{C} , \mathbf{l} , and \mathbf{p} **if 1** ($p < 0.5$) **if 2** ($|A| < +1$)

Update the current sol.'s Position

else if 2 ($|A| > +1$)

pick a random search agent

Modify the new search agent's location

end if 2 **else if 1** ($p \geq 0.5$)

Modify the latest search location

end if 1

end for

Verify that every solution reaches the search space and adjust it

Compute each solution's fitness

Modify X^* if a better solution occurs $t = t + 1$

end while

Obtain the optimal input weights and hidden layer biases of ELM by X^* ELM train and test*5.1.1 Classification Dataset*

Cancer is one of the most dreaded and violent illnesses in the world, causing over 9 million deaths. The earlier the level of cancer is, the better the chance of survival. The RNA sequence analysis is one of the steps [34]. There has been an interest in human genomics research recently in developing reliability and precision of artificial intelligence approaches and optimization algorithms. Centered on tumor RNA sequence (RNA-Seq) gene expression data, the authors present five hybrid models for classifying different forms of cancer. This study will focus on KIRC “kidney renal clear cell carcinoma”, BRCA “breast invasive carcinoma”, LUSC “lung squamous cell

carcinoma”, LUAD “lung adenocarcinoma”, and UCEC “uterine corpus endometrial carcinoma”. And the Description of the dataset will be illustrated in [Tab. 1](#).

5.1.2 Predication Dataset

With the proliferation of smart energy meters and the widespread deployment of electricity generation technologies such as solar panels, there is a wealth of data on electricity demand available [35].

This information is a multivariate time series of power-related variables that can be used to predict and estimate potential energy demand. This paper presents five hybrid models for predicting the household power consumption dataset, which explains energy demand for a single house over a four-year period, And the Description of the dataset will be illustrated in [Tab. 2](#).

Table 1: Gene expression cancer RNA-Seq dataset

| Name | No. of features | No. of samples |
|---|-----------------|----------------|
| Gene expression cancer RNA-seq data set | 16382 | 801 |

Table 2: Electricity load diagrams dataset description

| Name | No. of features | No. of samples |
|------------------------------------|-----------------|----------------|
| Electricity load diagrams data set | 140256 | 370 |

5.2 Parameters Settings

For the proposed and compared versions, 500 cycles have been planned. The ELM has 20 input layer nodes. It has twenty hidden nodes, but more hidden nodes are expected than algorithms of classical inclination. In the output layer, it has one node. The SSA, GOA, GWO, MFO and ELM algorithms parameter settings are summarized in the [Tab. 3](#).

5.3 Simulation Results

The following Results have been run on Google Colab, The VM used for Collaboratory appears to have 2-core Xeon 2.2GHz, 13GB RAM and 2 vCPU when checking using psutil (so a n1-highmem-2 instance)

5.3.1 Classification Results

The hybrid MFO-ELM model has a better accuracy than SSA-ELM, GOA-ELM GWO-ELM, WOA-ELM, standard ELM and SVM as shown.

1- Accuracy

MFO-ELM has a better Accuracy than other Algorithms as shown in [Tab. 4](#).

Table 3: Parameters settings

| Algorithm | Parameters | Value |
|---------------------------------|-----------------------|---------|
| SSA, GOA, GWO, MFO, and WOA ELM | Search agent's number | 20 |
| | Iterations number. | 100 |
| | Input nodes number | 20 |
| | Hidden nodes number | 20 |
| | Activation fun. | Sigmoid |
| | Output nodes number | 1 |
| | iterations number | 100 |
| SVM | Learning rate | 0.01 |
| | Input nodes number | 20 |
| | Output nodes number | 1 |
| | Iterations number | 50 |

Table 4: Accuracy of proposed models

| SSA-ELM | GOA-ELM | GWO-ELM | MFO-ELM | WOA-ELM | ELM | SVM |
|---------|---------|---------|---------|---------|-------|-------|
| 80.12 | 90.15 | 89.06 | 98.13 | 79.68 | 66.45 | 63.97 |

2- Learning Curves

All proposed models converged to global minimum after few iterations as shown in Figures from [Figs. 3–7](#).

3- Confusion Matrix

SSA-ELM has misclassified Class “COAD” between Class (COAD and BRCA) as shown in [Fig. 8](#). GOA-ELM has misclassified Class “COAD” as shown in [Fig. 9](#). GWO-ELM has misclassified Class “COAD” as shown in [Fig. 10](#). MFO-ELM has misclassified Class “PRAD” as shown in [Fig. 11](#). WOA -ELM has classified all classes correctly as shown in [Fig. 12](#). ELM has misclassified Class “COAD, KIRC” as shown in [Fig. 13](#).

4- Computational Time

Instead of the better accuracy if MFO- ELM, SVM has the best Computational Time, As shown in [Tab. 5](#), the time is calculated in Seconds

5- Precision

MFO- ELM has the best Precision across all classes as shown below in [Tab. 6](#).

6-Recall

MFO- ELM has the best Recall across all classes as shown below in [Tab. 7](#).

7- F1 Score

MFO- ELM has the best F1Score across all classes as shown below in [Tab. 8](#).

5.3.2 Prediction Results

The hybrid GOA-ELM model has a better accuracy than SSA-ELM, MFO-ELM GWO-ELM, WOA-ELM, standard ELM and SVM as shown.

1- Root Mean Square Error (RMSE):

GOA-ELM has the least Mean Squared Error as shown in Tab. 9.

2- Mean Absolute Error (MAE):

GOA-ELM has the least Mean Absolute Error as illustrated in Tab. 10.



Figure 3: SSA-ELM learning curve

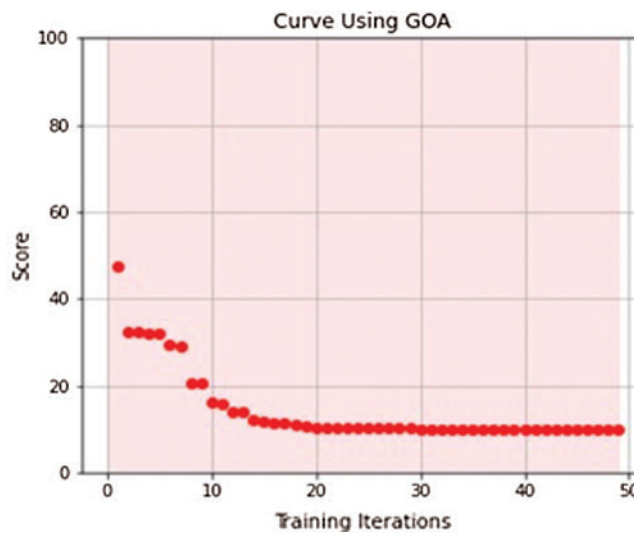


Figure 4: GOA-ELM learning curve

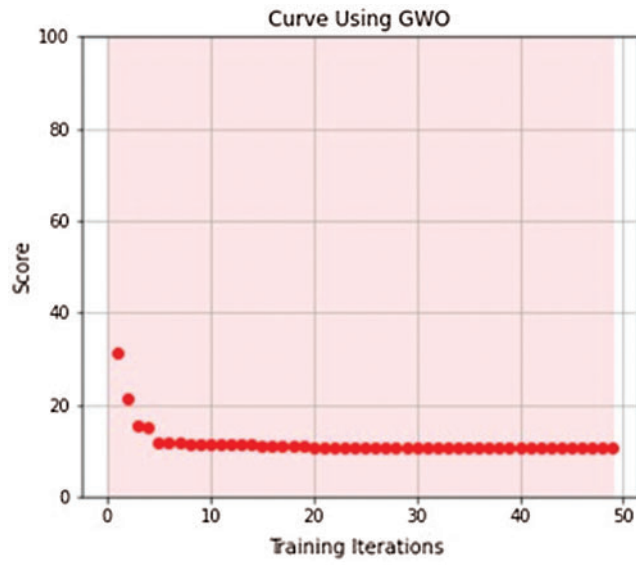


Figure 5: GWO_ELM learning curve

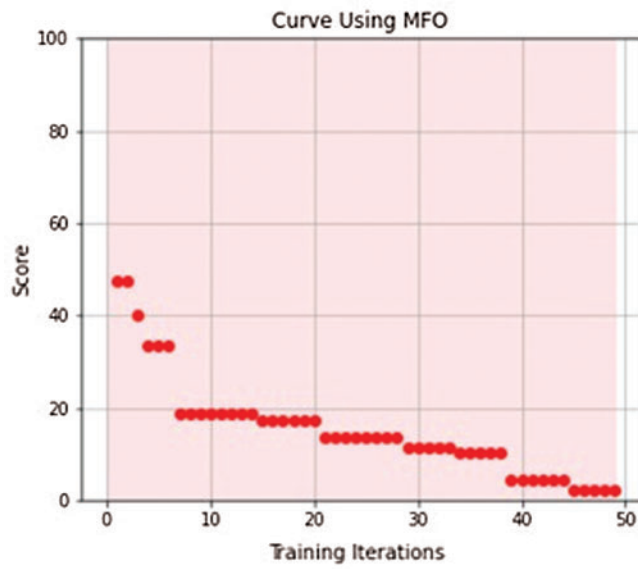


Figure 6: MFO-ELM learning curve

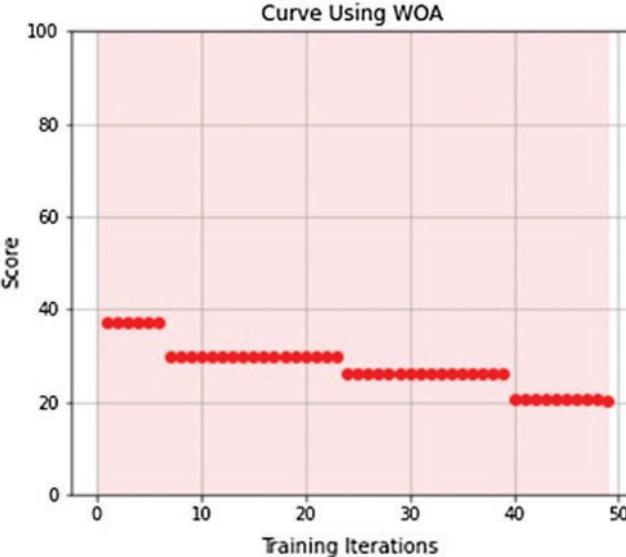


Figure 7: WOA-ELM learning curve

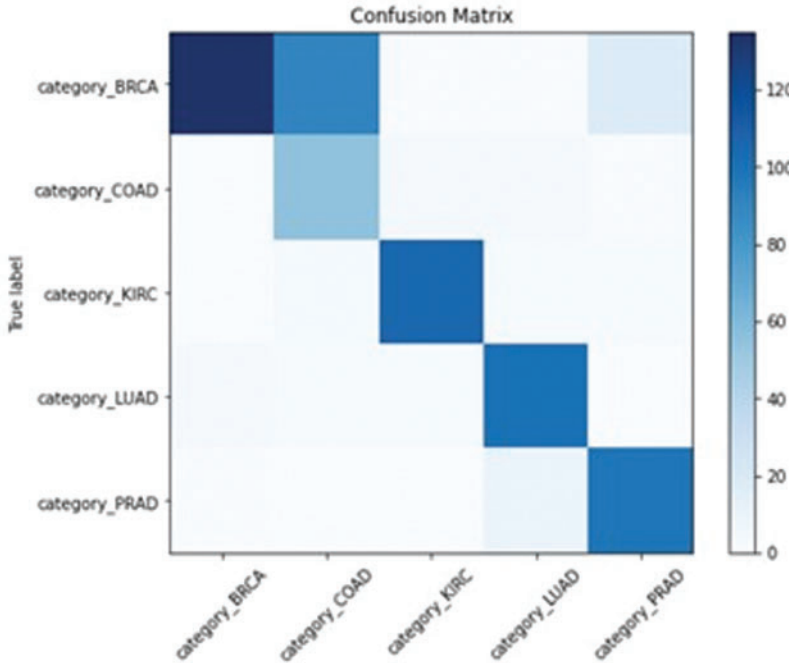


Figure 8: SSA-ELM confusion matrix

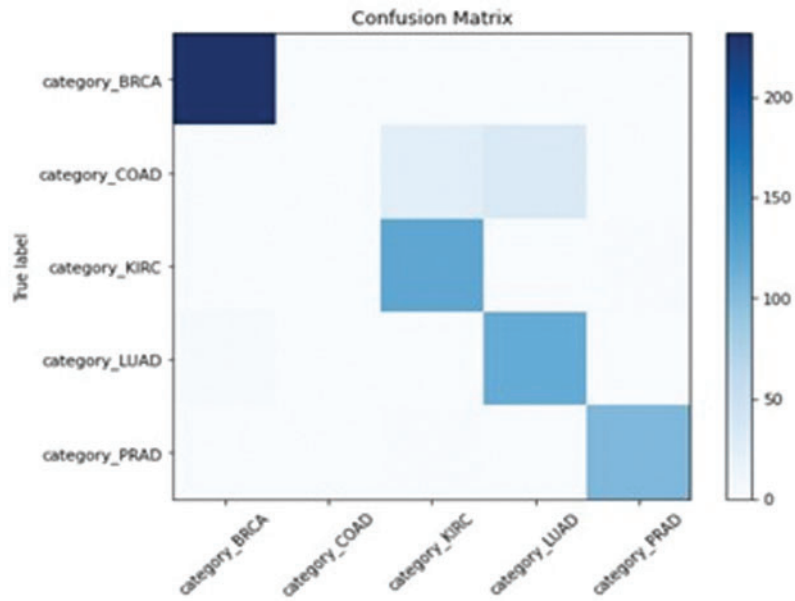


Figure 9: GOA-ELM confusion matrix

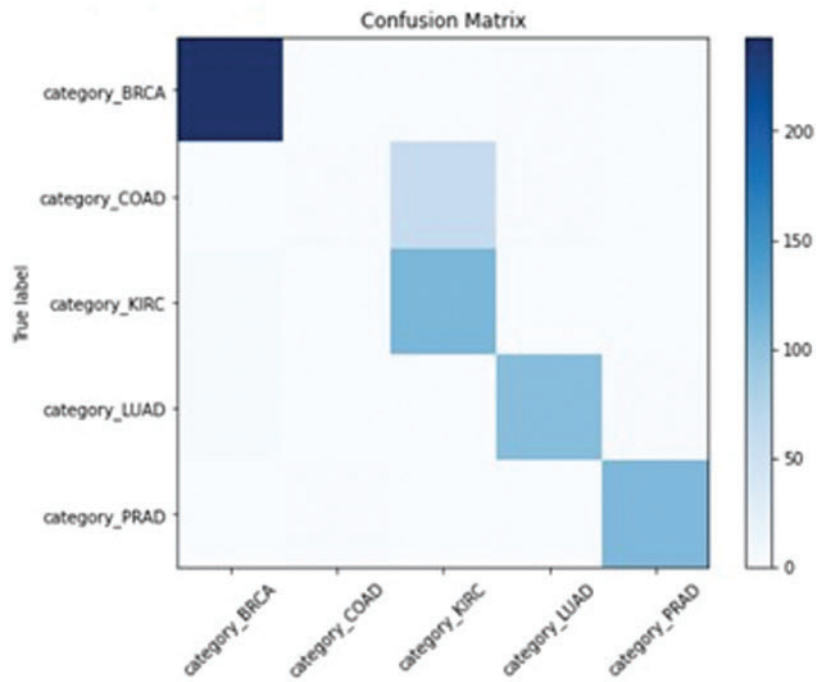


Figure 10: GWO-ELM confusion matrix

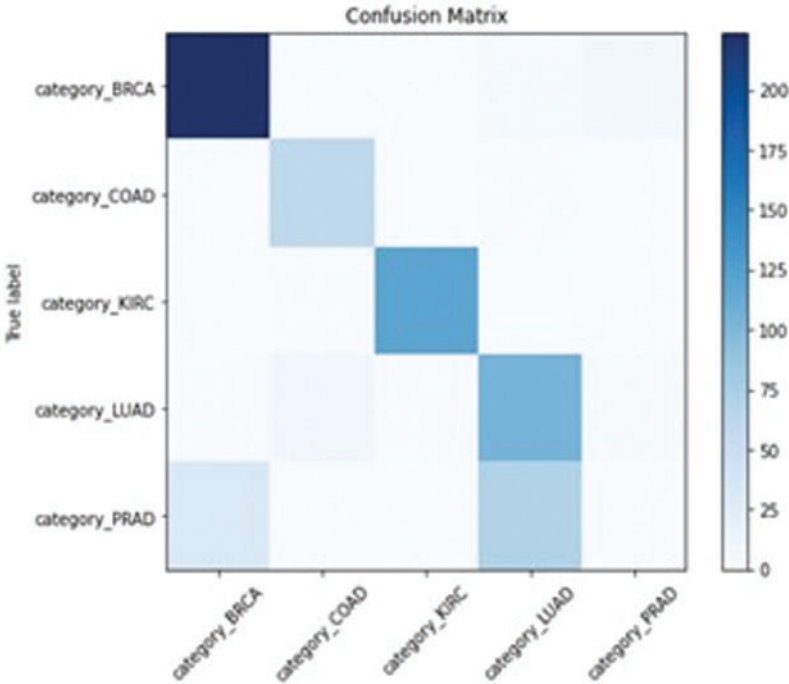


Figure 11: MFO-ELM confusion matrix

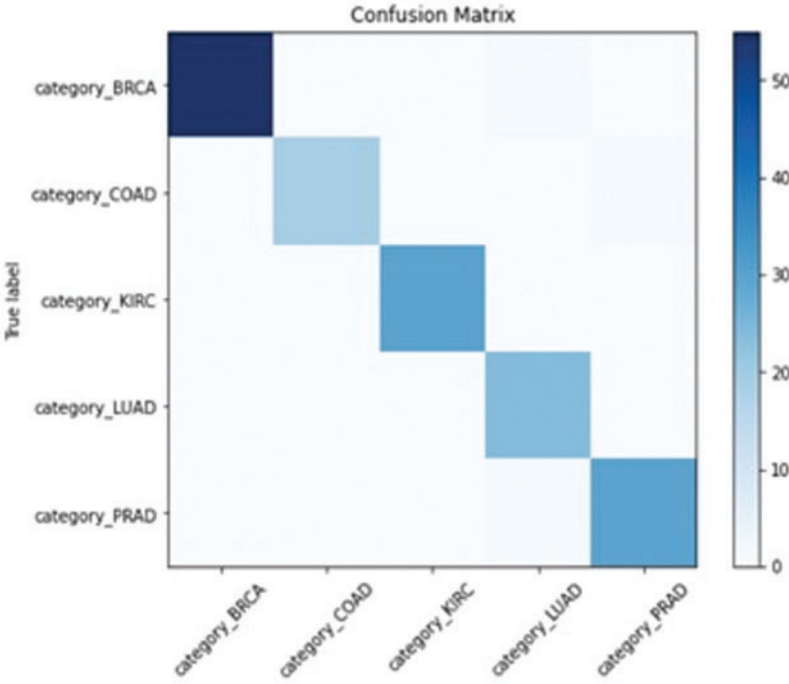


Figure 12: WOA-ELM confusion matrix

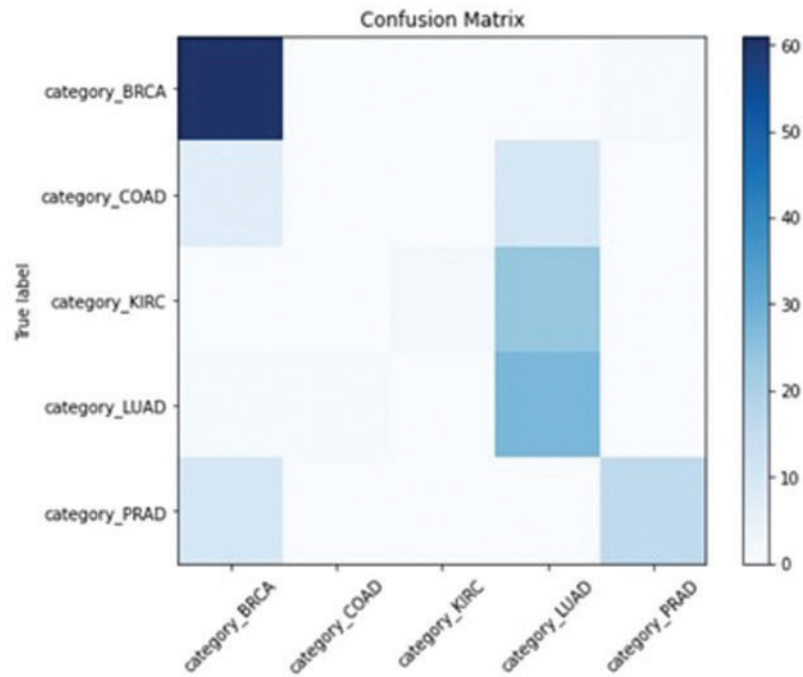


Figure 13: ELM confusion matrix

Table 5: Computational time

| SSA-ELM | GOA-ELM | GWO-ELM | MFO-ELM | WOA-ELM | ELM | SVM |
|---------|---------|---------|---------|---------|-------|-------|
| 15.817 | 17.006 | 16.392 | 15.974 | 16.405 | 0.357 | 0.055 |

Table 6: Precision

| | SSA-ELM | GOA-ELM | GWO-ELM | MFO-ELM | WOA-ELM | ELM | SVM |
|--------------|---------|---------|---------|---------|---------|------|------|
| PRAD | 0.94 | 0.99 | 0.96 | 1.00 | 0.82 | 0.77 | 0.98 |
| LUAD | 0.34 | 0.00 | 0.00 | 1.00 | 0.93 | 0.00 | 1.00 |
| KIRC | 0.94 | 0.83 | 0.67 | 1.00 | 0.83 | 1.00 | 1.00 |
| COAD | 0.88 | 0.77 | 0.94 | 0.92 | 0.44 | 0.45 | 0.00 |
| BRCA | 0.96 | 0.99 | 0.96 | 0.97 | 0.00 | 0.94 | 0.00 |
| Micro avg | 0.80 | 0.90 | 0.88 | 0.98 | 0.72 | 0.66 | 0.99 |
| Macro avg | 0.81 | 0.72 | 0.71 | 0.98 | 0.61 | 0.63 | 0.60 |
| weighted avg | 0.80 | 0.83 | 0.81 | 0.98 | 0.61 | 0.69 | 0.65 |

Table 7: Recall

| | SSA-ELM | GOA-ELM | GWO-ELM | MFO- ELM | WOA-ELM | ELM | SVM |
|--------------|---------|---------|---------|----------|---------|------|------|
| PRAD | 0.63 | 1.00 | 0.98 | 0.98 | 0.92 | 0.98 | 1.00 |
| LUAD | 0.86 | 0.00 | 0.00 | 0.95 | 0.88 | 0.00 | 0.88 |
| KIRC | 0.91 | 0.99 | 1.00 | 1.00 | 1.00 | 0.08 | 1.00 |
| COAD | 0.91 | 0.98 | 0.91 | 1.00 | 0.78 | 0.93 | 0.00 |
| BRCA | 0.86 | 0.99 | 1.00 | 0.97 | 0.00 | 0.62 | 0.00 |
| Micro avg | 0.80 | 0.90 | 0.88 | 0.98 | 0.72 | 0.66 | 0.64 |
| Macro avg | 0.83 | 0.79 | 0.78 | 0.98 | 0.72 | 0.52 | 0.58 |
| weighted avg | 0.80 | 0.90 | 0.88 | 0.98 | 0.72 | 0.66 | 0.64 |

Table 8: F1 Score

| | SSA-ELM | GOA-ELM | GWO-ELM | MFO-ELM | WOA-ELM | ELM | SVM |
|--------------|---------|---------|---------|---------|---------|------|------|
| PRAD | 0.76 | 1.00 | 0.97 | 0.99 | 0.87 | 0.87 | 0.99 |
| LUAD | 0.49 | 0.00 | 0.00 | 0.97 | 0.90 | 0.00 | 0.94 |
| KIRC | 0.92 | 0.90 | 0.81 | 1.00 | 0.91 | 0.14 | 1.00 |
| COAD | 0.89 | 0.87 | 0.92 | 0.96 | 0.56 | 0.61 | 0.00 |
| BRCA | 0.91 | 0.99 | 0.98 | 0.97 | 0.00 | 0.74 | 0.00 |
| Micro avg | 0.80 | 0.90 | 0.88 | 0.98 | 0.72 | 0.66 | 0.78 |
| Macro avg | 0.79 | 0.75 | 0.74 | 0.98 | 0.65 | 0.47 | 0.59 |
| weighted avg | 0.82 | 0.86 | 0.84 | 0.98 | 0.66 | 0.59 | 0.64 |

Table 9: Root mean square error (RMSE)

| SSA-ELM | GOA-ELM | GWO-ELM | MFO-ELM | WOAELM | ELM | SVM |
|---------|---------|---------|---------|--------|-------|-------|
| 0.449 | 0.397 | 0.400 | 0.463 | 0.642 | 0.660 | 0.422 |

Table 10: Mean absolute error (MAE)

| SSA-ELM | GOA-ELM | GWO-ELM | MFO-ELM | WOA-ELM | ELM | SVM |
|---------|---------|---------|---------|---------|-------|-------|
| 0.374 | 0.282 | 0.331 | 0.286 | 0.486 | 0.522 | 0.395 |

6 Conclusions and Future Work

This paper proposes five hybrid models for enhancing ELM efficiency. The proposed hybrid algorithms are namely, “Salp Swarm Algorithm (SSA-ELM), Grasshopper Algorithm (GOA-ELM), Grey Wolf Algorithm (GWO-ELM), Whale optimization Algorithm (WOA-ELM) and Moth Flame Optimization (MFO-ELM). Experimental results for gene classification dataset showed that MFO achieved best accuracy and efficiently select weights and hidden layer biases of ELM. The proposed models addressed the over-fitting issue that afflicted the regular ELM model. The five hybrid models had been used for predicting the household power consumption dataset, which explains energy demand for a single house over a four-year period. The proposed

GOA-ELM model has achieved the best result in predicting and estimating potential energy demand.

In Future work, New Hybrid models will be developed. in addition, we will try to test the hybrid models on more datasets in different fields and check the efficiency of each algorithm in each dataset and which fits each field. And we are going to try these models on Higher datasets with millions of features.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

- [1] Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten *et al.*, “Op-elm: Optimally pruned extreme learning machine,” *IEEE Transactions on Neural Networks*, vol. 21, no. 1, pp. 158–162, 2010.
- [2] G. B. Huang, Q. Y. Zhu and C. K. Siew, “Extreme learning machine: Theory and applications,” *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006.
- [3] F. Han and D. S. Huang, “Improved extreme learning machine for function approximation by encoding a priori information,” *Neurocomputing*, vol. 69, no. 1, pp. 2369–2373, 2006.
- [4] G. B. Huang, Q. Y. Zhu and C. K. Siew, “Extreme learning machine: A new learning scheme of feedforward neural networks,” in *Int. Joint Conf. on Neural Networks*, Budapest, Hungary, pp. 985–990, 2004.
- [5] X. Li, H. Xie, R. Wang, Y. Cai, J. Cao *et al.*, “Empirical analysis: Stock market prediction via extreme learning machine,” *Neural Computing and Applications*, vol. 27, no. 1, pp. 67–78, 2016.
- [6] Q. Y. Zhu, A. K. Qin, P. N. Suganthan and G. B. Huang, “Evolutionary extreme learning machine, pattern recognition,” *Pattern Recognition*, vol. 38, no. 10, pp. 1759–1763, 2005.
- [7] G. Zhao, Z. Shen, C. Miao and Z. Man, “On improving the conditioning of extreme learning machine: A linear case,” in *7th Int. Conf. on Information, Communications and Signal Processing*, Maucun, China, pp. 1–5, 2009.
- [8] Y. Wang, F. Cao and Y. Yuan, “A study on effectiveness of extreme learning machine,” *Neurocomputing*, vol. 74, no. 16, pp. 2483–2490, 2011.
- [9] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris *et al.*, “Salp swarm algorithm: A bio-inspired optimizer for engineering design problems,” *Advances in Engineering Software*, vol. 114, no. 1, pp. 163–191, 2017.
- [10] J. Luo, H. Chen, Y. Xu, H. Huang and X. Zhao, “An improved grasshopper optimization algorithm with application to financial stress prediction,” *Applied Mathematical Modelling*, vol. 64, no. 1, pp. 654–668, 2018.
- [11] V. Kumar, J. K. Chhabra and D. Kumar, “Grey wolf algorithm-based clustering technique,” *Journal of Intelligent Systems*, vol. 26, no. 1, pp. 153–168, 2017.
- [12] M. Shehab, L. Abualigah, H. Al Hamad, H. Alabool, M. Alshinwan *et al.*, “Moth–flame optimization algorithm: Variants and applications,” *Neural Computing and Applications*, vol. 32, no. 14, pp. 9859–9884, 2020.
- [13] S. Mirjalili and A. Lewis, “The whale optimization algorithm,” *Advances in Engineering Software*, vol. 95, no. 1, pp. 51–67, 2016.
- [14] M. F. Isham, M. S. Leong, M. H. Lim and Z. A. B. Ahmad, “Optimized ELM based on whale optimization algorithm for gearbox diagnosis,” *MATEC Web of Conferences, EDP Sciences*, vol. 255, no. 1, pp. 1–7, 2019.
- [15] G. B. Huang, Q. Y. Zhu and C. K. Siew, “Extreme learning machine, theory and applications,” *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

- [16] Q. HuaLing F. Han and H. FenYao, "An improved evolutionary extreme learning machine based on particle swarm optimization," *Neurocomputing*, vol. 116, no. 1, pp. 87–93, 2013.
- [17] M. Alhamdoosh and D. Wang, "Evolutionary extreme learning machine ensembles with size control," *Neurocomputing*, vol. 102, no. 1, pp. 98–110, 2013.
- [18] Z. Wu, J. Yang, X. Xue and M. Yao, "Genetic ensemble of extreme learning machine," *Neurocomputing*, vol. 129, no. 1, pp. 175–184, 2014.
- [19] N. Sundararajanc, S. Suresha and S. Saraswathib. "Performance enhancement of extreme learning machine for multi-category sparse data classification problems," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 7, pp. 1149–1157, 2010.
- [20] J. Zhao, Z. Dong, H. Yang and J. Yia, "Extreme learning machine based genetic algorithm and its application in power system economic dispatch," *Neurocomputing*, vol. 102, no. 7, pp. 154–162, 2013.
- [21] M. Abdul Salam, O. Hegazy and O. S. Soliman, "Fpa-elm model for stock market prediction," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 5, no. 2, pp. 1050–1063, 2015.
- [22] M. A. Salam, H. M. Zawbaa, E. Emary, K. K. A. Ghany and B. Parv, "A hybrid dragonfly algorithm with extreme learning machine for prediction," in *International Symposium on Innovations in Intelligent Systems and Applications (INISTA)*, Sinaia, Romania, pp. 1–6, 2016.
- [23] I. Aljarah, H. Faris and S. Mirjalili, "Optimizing connection weights in neural networks using the whale optimization algorithm," *Soft Computing*, vol. 22, no. 1, pp. 1–15, 2018.
- [24] P. Mohapatra, S. Chakravarty and P. K. Dash, "An improved cuckoo search based extreme learning machine for medical data classification," *Swarm and Evolutionary Computation*, vol. 24, no. 1, pp. 25–49, 2015.
- [25] X. Liu, L. Wang, G. B. Huang, J. hang and J. Yin, "Multiple kernel extreme learning machine," *Neurocomputing*, vol. 149, no. 1, pp. 253–264, 2015.
- [26] G. B. Huang, X. Ding and H. Zhou, "Optimization method based extreme learning machine for classification," *Neurocomputing*, vol. 74, no. 1-3, pp. 155–163, 2010.
- [27] W. Zong, G. B. Huang and Y. Chen, "Weighted extreme learning machine for imbalance learning," *Neurocomputing*, vol. 101, no. 1, pp. 229–242, 2013.
- [28] N. Sowmya and V. Ponnusamy, "Development of spectroscopic sensor system for an IoT application of adulteration identification on milk using machine learning," *IEEE Access*, vol. 9, no. 1, pp. 53979–53995, 2021.
- [29] S. Ding, H. Zhao, Y. Zhang, X. Xu and R. Nie, "Extreme learning machine: Algorithm, theory and applications," *Artificial Intelligence Review*, vol. 44, no. 1, pp. 103–115, 2015.
- [30] S. Mirjalili, S. M. Mirjalili and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, no. 1, pp. 46–61, 2014.
- [31] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowledge-Based Systems*, vol. 89, no. 1, pp. 228–249, 2015.
- [32] L. Tan, J. Han and H. Zhang, "Ultra-short-term wind power prediction by salp swarm algorithm-based optimizing extreme learning machine," *IEEE Access*, vol. 8, no. 1, pp. 44470–44484, 2020.
- [33] H. Faris, S. Mirjalili, I. Aljarah, M. Mafarja and A. A. Heidari, "Salp swarm algorithm: Theory, literature review, and application in extreme learning machines," *Nature-Inspired Optimizers*, vol. 811, pp. 185–199, 2020.
- [34] K. Bache and M. Lichman, "Uci machine learning repository," 2013. [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/gene+expression+cancer+RNA-Seq>.
- [35] K. Bache and M. Lichman, "Uci machine learning repository," 2013. [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014>.