Aquila Optimizer for hyperparameter metaheuristic optimization in ELM

Philip Vasquez-Iglesias[1], David Zabala-Blanco[1], Amelia E. Pizarro[1], Juan Fuentes-Concha[1], and Paulo Gonzalez[2]

[1] Facultad de Ciencias de la Ingeniería, Universidad Católica del Maule, Chile
{fvasquez,dzabala}@ucm.cl
amelia.pizarro.riquelme@gmail.com
juan.fuentes.01@alumnos.ucm.cl
[2] Facultad de Economía y Negocios, Universidad de Talca, Talca, Chile
paulo.gonzalezg@utalca.cl

Abstract. This paper introduces the adaptation of the Aquila Optimizer (AO) metaheuristic for optimizing the hyperparameters of the Extreme Learning Machine (ELM). The AO algorithm is a swarm intelligence-based metaheuristic that optimizes the objective function by simulating aquilas hunting behavior. The ELM belongs to the family of single hidden layer feed-forward network algorithms, where the hidden layer weights are randomly initialized and whose training is based on the Moore-Penrose pseudoinverse. It is known for faster convergence than traditional methods, providing promising performance with minimal programmer intervention. The proposed method focuses on optimizing the hidden neurons of the ELM by maximizing the most popular performance metrics, namely Accuracy and G-Mean. This method offers an alternative to the classic grid search method by avoiding the need to go through all possible combinations in search of the optimal value. We evaluated three typical datasets and found that our proposal achieves an average efficiency of 90% compared to the global maximum found by the grid search, reducing the search time by 25%. In other words, our method can achieve performance close to the global maximum in a fraction of the time required by the brute force methodology.

Keywords: Aquila Optimizer · Heuristic Parametrization · Extreme Learning Machine · Soft Computing.

1 Introduction

Optimization involves finding the best solution by evaluating a given objective function at one or more values within a specified domain. Different optimization methods, such as classical methods based on the derivative information like gradient descent [2]. On the other hand, metaheuristic methods do not rely on derivative information; instead, they use heuristics to find the best solutions.

Evolutionary optimization methods are a branch of Soft Computing that take inspiration from nature to mimic behaviors that have allowed many species

to survive. These methods are motivated by the behavior of ant colonies, bee colonies, and fish schools. They offer the following advantages over traditional optimization methods: (1) Do not require differentiability, and (2) Tend to be more resistant to getting stuck in local optimal solutions in complex, multimodal functions. Some well-known algorithms in this category include Genetic Algorithms (GA) [5], Particle Swarm Optimization (PSO) [8], and Artificial Bee Colony (ABC) [7].

In metaheuristic optimization, the parameterization is one problem that can be addressed. The following steps are required to obtain it: (1) Identifying an objective function, typically a performance function that determines whether to maximize or minimize; and (2) Determining the necessary constraints, which usually involve different parameters and their valid domains. In [4], the Anisotropic diffusion filter of Perona and Malik [11] was parameterized using Karaboga's Artificial Bee Colony algorithm [7] by adjusting the filter parameters (numOfIteration, scaleParameter, and timeStep) and performance functions to measure the quality of an image (Signal to Noise Ratio (SNR) and Mean Squared Error (MSE)). In the ongoing research, the AO algorithm is employed to optimize the hyperparameter corresponding to the number of neurons in the hidden layer of a neural network model known as an Extreme Learning Machine on various data sets.

The ELM is widely used to resolve various classification problems. It is becoming an exciting alternative to health-related problems due to its good performance without the need to invest time in training. In this paper, we present a method capable of achieving a performance close to the best value found by an exhaustive search, limiting the run instances of the ELM done by AO to approximately 25% of the ELM executions done by the exhaustive search.

This paper is divided into five sections. Section 1 introduces metaheuristic optimization. Section 2 presents the base knowledge of AO and ELM algorithms to understand the terminology used. Section 3 describes the proposed method for using AO metaheuristic optimization over ELM networks. Section 4 presents the results achieved by ELM across several datasets using both exhaustive search and the AO algorithm as hyperparameter optimization methods, and their performance is discussed. Finally, Section 5 summarizes the conclusions of the proposed work.

2 Theoretical Framework

2.1 Aquila Optimizer Algorithm

The Aquila Optimizer (AO) algorithm, introduced in [1], is based on the hunting behavior of aquilas. The algorithm begins with a random distribution of agents (aquilas) in the search space, and the quality of each location is assessed using an objective function. The obtained information is stored and used by the rest of the agents to generate new solutions in an iterative process. The best solution discovered is periodically reported until the final iteration of the algorithm, which depends on a predetermined maximum number of iterations.

The hunting options modeled in the AO include (1) High vertical flight to select the search space (The Expanded Exploration), (2) Short glide attack, allowing divergent exploration of the search space (Narrow Exploration), (3) Slow descent attack to exploit the convergent search space (Expanded exploitation), and (4) Ground attack to capture prey (Narrowed exploitation). It is important to note that options 1 and 2 are exploration or coarse search strategies, while alternatives 3 and 4 are exploitation or refined search strategies.

In the iterative process, each agent will be relocated using one of four hunting options based on the progress criterion, which depends on the percentage of iterations completed. Let us say T is the total number of iterations for each agent in each cycle t. If $t \leq \frac{2}{3} \times T$, the new position for the agent will be determined using an exploration criterion from either Equation 1 or 2, chosen randomly by following an uniform distribution. If this situation is not met, one-third of the total number of cycles remains. Therefore, the new position for the agent will be determined using an exploitation criterion from either Equation 7 or 8, also randomly selected.

In Equation 1, the first kind of attack, The Expanded Exploration, is modeled.

$$X_{\text{New}}(t+1) = X_{\text{Best}}(t) \times \left(1 - \frac{t}{T}\right) + \left(X_{\text{mean}}(t) - X_{\text{Best}}(t) \times \text{rand}(\cdot)\right), \quad (1)$$

such that $X_{\text{new}}(t+1)$ corresponds to the solution of the next iteration to the current t, $X_{\text{Best}}(t)$ is the best position found in the current iteration, $X_{\text{mean}}(t)$ is the average among all positions of the current iteration and $rand(\cdot)$ comes to be a random number between 0 and 1.

In Equation 2, the second kind of attack, Narrow Exploration, is modeled.

$$X_{\text{New}}(t+1) = X_{\text{Best}}(t) \times Levy(D) + X_{\text{R}}(t) + (y - x) \times rand(\cdot), \quad (2)$$

such that, D is dimension space, $X_{\text{R}}(t)$ is a random solution taken between 1 and N at the i^{th} iteration, and $Levy(D)$ is the Levy flight distribution function, calculated using Equation 3.

$$Levy(D) = s \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}}, \quad (3)$$

such that s is a constant value equal to 0.01, u and v are random numbers between 0 and 1, β is a constant value equal to 1.5, and σ is a number calculated using Equation 4.

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}} \right). \quad (4)$$

In Equation 2, y and x are used to present the spiral in the search, calculated using Equations 5 and 6, respectively.

$$y = (r_1 + U \times D_1) \times \cos\left(-\omega \times D_1 + \frac{3 \times \pi}{2}\right) \tag{5}$$

$$x = (r_1 + U \times D_1) \times \sin\left(-\omega \times D_1 + \frac{3 \times \pi}{2}\right), \tag{6}$$

such that r_1 is a value equal to several iterations, U is a constant value equal to 0.00565, D_1 is an array with the numbers from 1 to the length of the search space (Dim), and ω is a constant value equal to 0.005.

In Equation 7, the third kind of attack, Expanded exploitation, is modeled.

$$X_{\text{New}}(t+1) = (X_{\text{Best}}(t) - X_{\text{mean}}(t)) \times \alpha - rand(\cdot) + ((UB - LB) \times rand(\cdot) + LB) \times \delta, \tag{7}$$

such that, α and δ are the exploitation adjustment parameters fixed in the original paper to 0.1, LB and UB denotes to the lower and upper bound, respectively.

In Equation 8, the fourth kind of attack, Narrowed exploitation, is modeled.

$$X_{\text{New}}(t+1) = QF(t) \times (X_{\text{Best}}(t) - (G_1 \times X(t) \times rand(\cdot)) - G_2 \times Levy(D) + rand(\cdot) \times G_1, \tag{8}$$

such that $QF(t)$ is the quality function used to equilibrium the search strategies in the current iteration, which is calculated using Equation 11. G_1 denotes the motions of the Aquila that are used to track the prey, is calculated by Equation 11. G_2 is a decreasing value from 2 to 0, denote the flight slope of the Aquila that is used to follow the prey during the slope from the first location to t^{th} location, which is calculated using Equation 11.

$$QF(t) = t^{\frac{2 \times rand(\cdot) - 1}{(1-T)^2}}, \tag{9}$$

$$G_1 = 2 \times rand(\cdot) - 1, \tag{10}$$

$$G_2 = 2 \times \left(1 - \frac{t}{T}\right), \tag{11}$$

The Expanded Exploration (1) and Narrow Exploration (2) equations allow a coarse search in the space exploration, while the equations Expanded exploitation (7) and Narrowed exploitation (8) allow a refined search in the space exploitation.

The component that makes AO attractive is its robust mathematical foundation and low computational cost relative to other swarm intelligence algorithms. In [1], an extensive performance comparison is reported between the AO

and other metaheuristic algorithms applied to the optimization of fundamental problems and 23 classical mathematical functions of recurrent use in the literature, classifiable in different categories and difficulties (unimodal, multimodal and fixed-dimensional multimodal). The study reports a favorable result for AO. In [3], an improvement of the PSO algorithm of Kennedy is proposed [8], where very similar results are reported with the AO, by leaving behind other algorithms such as the original PSO and the ABC of Karaboga [7].

2.2 Extreme Learning Machine

The Extreme Learning Machine (ELM) is a type of neural network characterized by a hidden layer with randomly initialized weights [6]. It is part of the family of random weights neural networks, eliminating the need to adjust the weights of the hidden layer model iteratively. As a result, the training time for neural networks is significantly reduced compared to the gradient descent method. It is worth noting that other similar methods, such as Random Vector Functional Link (RVFL) [9] and those discussed in [12], have been proposed with slight differences, mainly related to residual connections.

Figure 1 provides a visual representation of the ELM algorithm's basic structure. Here, $\mathbf{X}_{d\times N}$ denotes the dataset, a collection of N samples each with d features. The diagram also includes $\tilde{N}$ as the number of neurons in the hidden layer, m the number of neurons in the output layer, $\mathbf{b}$ the biases of the hidden layer neurons (which are randomly generated), $\mathbf{H}$ the output of the hidden layer, $\boldsymbol{\beta}$ the weights of the hidden layer, $\mathbf{T}$ the expected output, and $\mathbf{W}$ the weights between the input and hidden layer (also randomly generated).

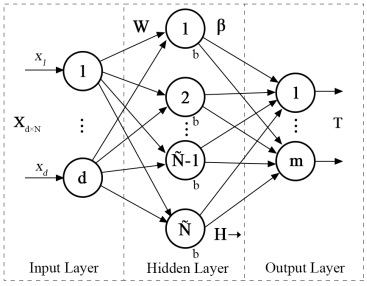


Fig. 1. ELM model.

The training of the network consists of the search for the least squares solution $\hat{\boldsymbol{\beta}}$ of the linear system given by $\mathbf{H}\boldsymbol{\beta} = \mathbf{T}$. If the amount of training data N and the number of neurons $\tilde{N}$ are equal, the solution will be exact with $\mathbf{H}$ square matrix and invertible, fitting the training data without error. In practice, these values differ, so $\mathbf{H}$ will correspond to a rectangular matrix and, therefore, no inverse. This results in the above system of equations containing multiple

solutions. According to [6], one of these solutions is given by:

$$\hat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\mathbf{T}, \tag{12}$$

where $\mathbf{H}^{\dagger}$ corresponds to the Moore-Penrose pseudoinverse. This solution is optimal because it minimizes the network's error and norm weights [6].

3 Proposed method

The proposed method consists of adapting the AO algorithm to optimize the hyperparameters of a neural network. In this work, AO is utilized to find the optimum number of neurons in an ELM by maximizing an objective function that reflects its performance.

It is necessary to modify the equations representing the aquila attacks presented in the previous chapter (1, 2, 7 y 8) by incorporating a rounding process because the solutions are indexes that allow access to the values of the vectors containing the valid domain for each dimension.

The cardinality of the entire space search is denoted by EB and is calculated as shown in equation 13.

$$EB = \prod_{i=1}^{Dim} \#EB_i, \tag{13}$$

such that $\#EB_i$ is the cardinality of the i^{th} Space Search, where each i^{th} cardinality is calculated based on the domain of the i^{th} dimension.

In order to define the restriction of the Space Search's domain of the number of neurons, its cardinality is represented by $EB_{N^{o}Neural}$ and is computed according to the equation 14, where $N\_samples$ is the number of samples in the dataset, and $STEP$ is the step size. P is a percentage equal to 80% of the number of samples present in the used dataset, meaning that a dataset with 1000 samples is processed by an ELM with a domain in the interval $[1, 800]$.

$$EB_{N^{o}Neural} = \left\lceil \frac{P \times N\_samples}{STEP} \right\rceil. \tag{14}$$

In the AO algorithm, two parameters can affect the obtained results: the number of aquilas and the number of iterations. The number of aquilas is typically set to 10 based on previous literature. The number of iterations is determined using the following equation 15, where $Perc$ is a user-defined value between 1 and 100 representing the approximate percentage of time for an exhaustive search in $EB_{N^{o}Neural}$, and N is the number of aquilas set as a parameter in the method.

$$T = \left\lceil \frac{Perc \times EB}{100 \times N} \right\rceil \tag{15}$$

From the clearance of expression 15, it is possible to determine the total number of trips made by all aquilas within the same configuration, resulting in equation 16. This number is equivalent to the total number of ELM runs

performed by AO, while the number of ELM runs in each iteration of AO is equal to N.

$$Total\_Travels = T \times N = \left\lceil \frac{Perc \times EB}{100} \right\rceil. \tag{16}$$

To use an objective function that allows numerically measuring the neural network's performance in different situations and to apply optimization methods such as the proposed one, two metrics (accuracy and G-Mean) are used together with the average, both of which are fundamental in evaluating the classification methods.

Accuracy measures the overall prediction rate of the classifiers, quantifying the accuracy with which the models can classify the samples as a whole. It is from the raw comparison between the value obtained from the network and the desired value. We obtained the accuracy from the equation 17.

$$\text{Accuracy} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(t_i - \bar{t_i})^2}, \tag{17}$$

where N represents the number of samples, t_i and $\bar{t_i}$ denote the actual and predicted values, respectively.

The G-Mean is a measure that focuses on evaluating the specific prediction rate of classifiers, especially in the case of unbalanced classes. From the product, we obtain the accuracy of each class. The G-Mean considers the success of each class, including the minority ones, equally important, making it a more representative metric. According to [10], we use the equation 18 to obtain the G-Mean.

$$\text{G-Mean} = \sqrt[L]{\beta_1\beta_2...\beta_L}, \tag{18}$$

where L corresponds to the number of classes and $beta_i$ the accuracy in the i^{th} class.

In addition to the previously described metrics, this work considers pertinent the optimization of the average of both from a linear weighting for each, as shown in equation 19.

$$\text{Ponderated} = \frac{Accuracy + \text{G-Mean}}{2}. \tag{19}$$

Once the aforementioned adaptations have been made, the AO algorithm executes in each iteration a number of ELMs equal to the number of agents established. It receives as parameters the training and testing sets and the value of the hyperparameter associated with the position in which the agent is located. From this configuration, the neural network is trained, evaluates its performance and returns as a result a number between 0 and 1 belonging to one of the previously described metrics (G-Mean, Accuracy or Ponderated). AO uses this value as a quality criterion of the index content and will be used to generate new solutions in the following iterations from the equations 1, 2, 7 or 8, as appropriate.

During the executions, the information on the content of the index that reports the highest performance is stored. This value, which represents the best solution found by the AO algorithm, will be the final output of the algorithm.

4 Discussion and results

We use the UCI Heart Disease Dataset, Diabetes Dataset, and UCI Cardiotocography 3-Class Dataset because all of them are related to health problems and have different characteristics and levels of difficulty. In contrast, the first one is a balanced binary class problem. The second is a unbalanced binary class problem, and the third is a unbalanced 3-class problem. In each case, we determine the number of iterations of AO using T (Equation 15), with approximately 25% run percentage ($Perc$) and ten aquilas ($N = 10$). In order to compare with the proposed method, we conducted an exhaustive search with a cardinality of $EB = EB_{N^{o}Neural}$ and $STEP$ equal to 1. The number of executions of ELM done by the exhaustive search is equal to EB, while the number of executions of ELM done by AO is calculated by multiplying $T \times N$.

The metrics employed to assess ELM performance are *Accuracy*, G-Mean and *Ponderated*, calculated using the formulas 17, 18, and 19, as described in the previous section. In all experiments, the datasets were standardized to values between -1 and 1 to prevent any single feature from dominating the model, and a stratified k-fold cross-validation was performed to prevent overfitting. Each of the three metrics was used independently as an objective function for the AO experiments. In each experiment, the results of all three metrics were reported, regardless of which one was used as the objective function. Three graphics are presented for each dataset, figures 2, 4, and 6, by showing the result of the exhaustive search. Figures 3(a), 5(a), and 7(a) depict the results of AO by adopting each metric as an objective function. The full dot represents the point where the superior value of the objective function was reached, while the empty dot represents the best value of the metrics when they were not the objective function; the segmented lines demonstrate the best value found in the exhaustive search for each metric as a reference case. Finally, figures 3(b), 5(b) and 7(b) show the AO results together, where each metric is identified with a different color and each objective function with a particular line type, by highlighting the best value found in each curve via the following markers: dot, cross or asterisk.

4.1 UCI Heart Disease Dataset Results

This dataset contains 1025 samples characterized by 12 inputs and a single output (binary classification). The class represents either the absence or presence of disease, having 499 and 526 samples, respectively. Namely, the classification problem is balanced. For these experiments, $EB = 820$, $T = 21$, and $T \times N = 210$, inferring that the AO occupies 25.61% runs of the force brute ELM.

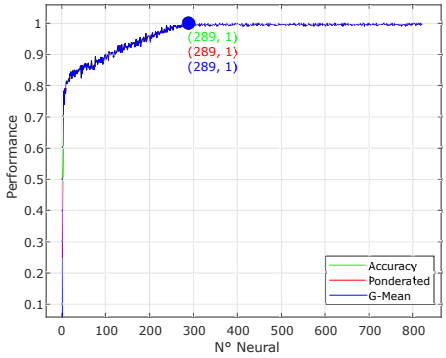
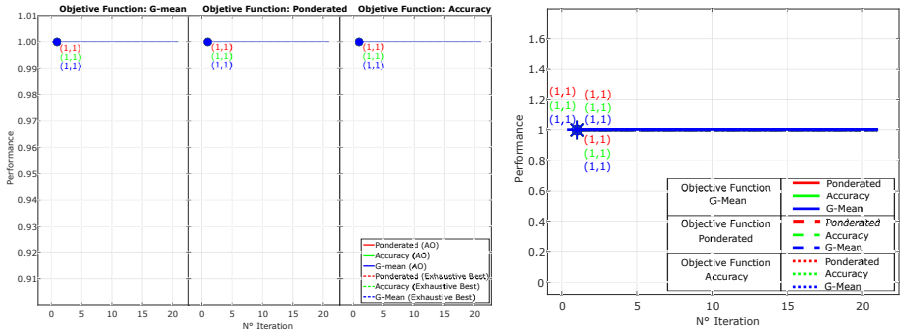


Fig. 2. Performance in terms of the ELM hidden neurons for the UCI Heart Disease Dataset and the exhaustive search approach.

In Figure 2, it is evident that the optimal point is reached at 289 hidden neurons, achieving 100% in the three metrics used as the objective function. The results remain consistent beyond this number of neurons.



(a) Results for the AO by comparing with the exhaustive search values.

(b) Summary for the nine results among the three objective functions.

Fig. 3. Results and experiments summarization of AO for UCI Heart Disease Dataset.

Figures 3(a) and 3(b) show that AO finds the best solution in the first iteration. In this case, the robustness of AO is still not exploited because, in the first iteration, AO has performed just a random search. In any case, the UCI Heart Disease Dataset is an easy problem for ELM. If we assume that after 288 neurons, the performance of the metrics is always 1, each of the ten aquilas has a high possibility of 63.3875% to randomly select several neurons over 288. Consequently, AO has a possibility in the first iteration of approximately 99.97% to

have at least one aquila with a neuron over 288, resulting in a performance of 100% in the metrics.

In general, the UCI Heart Disease Dataset results show that it is an easy problem for an ELM to tackle. Likewise, using AO for the ELM hyperparameter optimization allows for the easy finding of the optimal value due to the behavior of ELM in this dataset, independent of the objective function.

4.2 Diabetes Dataset Results

This dataset comprises a total of 768 samples, each with 9 attributes and a binary class. Class 0, indicating the absence of diabetes, is represented by 505 samples, while class 1, signifying its presence, is represented by 263 samples, making it an unbalanced problem that we aim to address.

For these experiments, $EB = 615$, $T = 16$, and $T = 160$, this implies that AO perform 26.02% of the total ELM runs done by the exhaustive search.

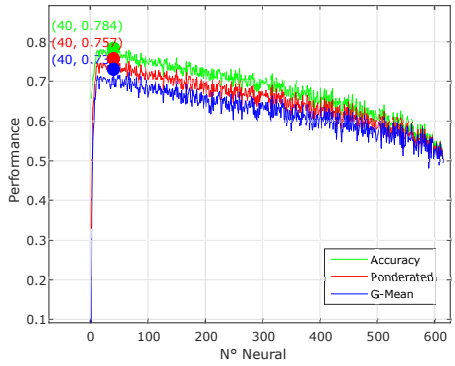
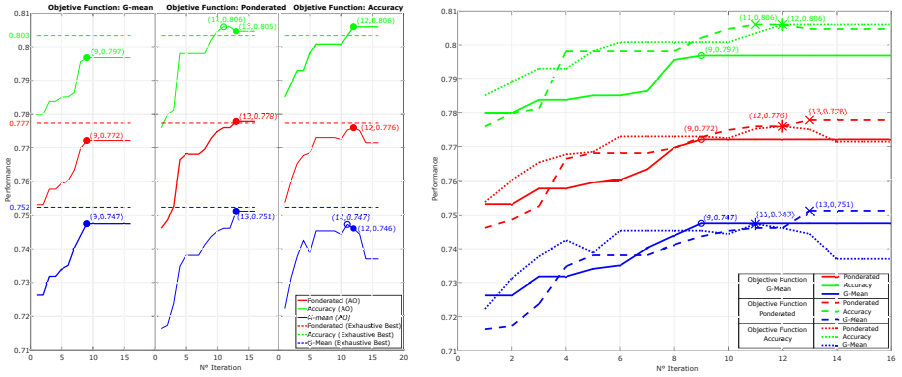


Fig. 4. Performance in terms of the ELM hidden neurons for the Diabetes Dataset and the exhaustive search approach.

Figure 4 shows a steady increase in the value of the metrics until the optimal value is reached using 40 neurons. After this point, the performance of each metric begins to decay.

Figure 5(a) reveals a behavior that consistently increases across all metrics, regardless of the objective function used. This behavior tends to approach the best value found in the exhaustive search and, in some cases, even surpasses it. Notably, the most superior results were achieved when the Ponderated metric was used as the objective function, with an Accuracy value that was only marginally lower than when the same Accuracy was used as the objective function.

Figure 5(b) shows that the results for the same metric across the different objective functions tend to cluster together, with all the results for *Accuracy* being at the top, followed by *Ponderated* and $G - Mean$.



(a) Results for the AO by comparing with the exhaustive search values.

(b) Summary for the nine results among the three objective functions.

Fig. 5. Results and experiments summarization of AO for Diabetes Dataset.

Interestingly, the use of Accuracy as the objective function does not always lead to a good G-Mean result. Conversely, using G-Mean as the objective function does not necessarily decrease the value of Accuracy. In this dataset, the most intriguing finding is that the best G-Mean value was obtained when using the ponderated metric as the objective function.

4.3 UCI Cardiotocography 3-Class Dataset Results

This dataset has 2126 samples, 22 attributes, and three classes. The class represents the fetal phase, where N is normal with 1655 samples, S is suspicious with 295 samples, and P is pathologic with 176, representing an unbalanced classification problem.

For these experiments, $EB = 1071$, $T = 43$, and $T \times N = 430$, implying that AO performs 25.28% of the ELM runs in the exhaustive search.

Figure 6 shows a steady increase in their performance until they reach their optimal value and then start to decay. It should be emphasized that the metrics have similar behavior but with different numbers of neurons at the optimal value for each metric.

Figure 7(a) shows that in this unbalanced multiclass problem, it is essential to consider G-Mean in the objective function, either when it is used or part of another, as in the Ponderated metric.

Figure 7(b) shows that the results for the same metric in the different objective functions tend to cluster together, just as in the previous dataset. In particular, the effect of using Accuracy as the objective function is seen, being itself at the top of the graph, while G-Mean is at the bottom.

The results show the need to not neglect G-Mean in unbalanced multiclass problems. When Accuracy is used as the objective function makes that the network tend to be optimized based on the majority class. finally, using Ponderated

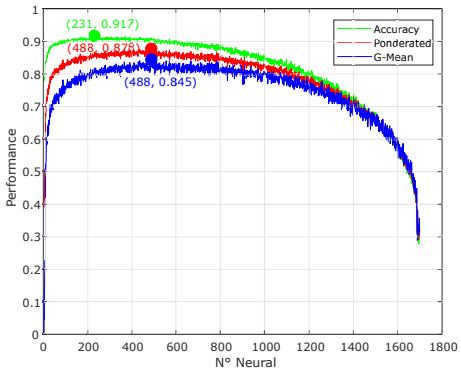
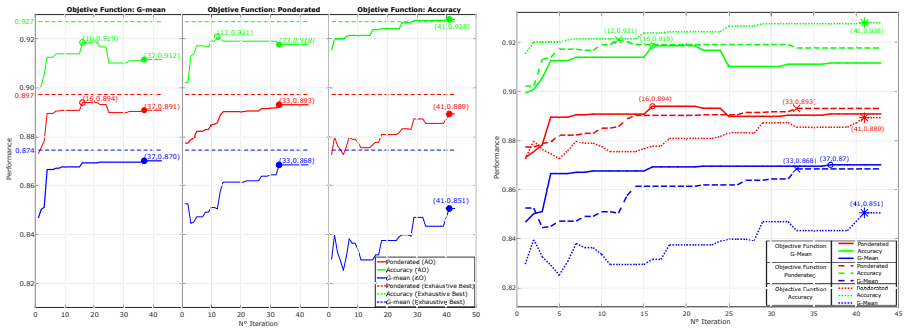


Fig. 6. Performance in terms of the ELM hidden neurons for the UCI Cardiotocography 3-Class Dataset and the exhaustive search approach.



(a) Results for the AO by comparing with the exhaustive search values.

(b) Summary for the nine results among the three objective functions.

Fig. 7. Results and experiments summarization of AO for UCI Cardiotocography 3-Class Dataset.

as the objective function generated a good balance in the values of G-Mean and Accuracy, maintaining the performance of G-Mean just 0.002 lower than when G-Mean is used as the objective function, while the value of Accuracy is still good.

4.4 Performance of AO respect to the Exhaustive Search

Table 1 shows a summary of the results obtained categorized by dataset. the Dataset column shows the name of the dataset with the type of experiment and the used metric, the Time (seconds) columns is the mean of time and the respective standard deviation (STD). Finally, the Performance column show the mean of the results of each experiment and they STD.

Table 1. Experiment summary table.

Dataset	Time (seconds)	Performance
UCI Heart Disease Exh(G-Mean)	61.081 ± 4.560	1.000 ± 0.000
UCI Heart Disease AO(G-Mean)	19.627 ± 3.344	1.000 ± 0.000
UCI Heart Disease Exh(Ponderated)	61.081 ± 4.560	1.000 ± 0.000
UCI Heart Disease AO(Ponderated)	21.985 ± 1.620	1.000 ± 0.000
UCI Heart Disease Exh(Accuracy)	61.081 ± 4.560	1.000 ± 0.000
UCI Heart Disease AO(Accuracy)	20.934 ± 4.869	1.000 ± 0.000
Diabetes Exh(G-Mean)	28.407 ± 0.565	0.752 ± 0.019
Diabetes AO(G-Mean)	3.128 ± 1.787	0.747 ± 0.021
Diabetes Exh(Ponderated)	28.407 ± 0.565	0.777 ± 0.014
Diabetes AO(Ponderated)	3.429 ± 1.751	0.778 ± 0.006
Diabetes Exh(Accuracy)	28.407 ± 0.565	0.803 ± 0.013
Diabetes AO(Accuracy)	4.113 ± 0.864	0.806 ± 0.021
UCI Cardiotocography 3-Class Exh(G-Mean)	987.997 ± 9.863	0.874 ± 0.012
UCI Cardiotocography 3-Class AO(G-Mean)	88.915 ± 38.111	0.870 ± 0.019
UCI Cardiotocography 3-Class Exh(Ponderated)	987.997 ± 9.863	0.897 ± 0.009
UCI Cardiotocography 3-Class AO(Ponderated)	95.299 ± 47.952	0.893 ± 0.010
UCI Cardiotocography 3-Class Exh(Accuracy)	987.997 ± 9.863	0.927 ± 0.009
UCI Cardiotocography 3-Class AO(Accuracy)	114.438 ± 34.707	0.928 ± 0.011

Based on Table 2, it can be concluded that the proposed method allows for the efficient and effective optimization of the hyperparameters of an ELM. The results obtained always remained above 99% of the average maximum value found by the exhaustive search, with average time percentages of 10.08% in the UCI Cardiotocography 3-Class Dataset, 12.52% in the Diabetes Dataset, and 34.13% in the UCI Heart Disease Dataset, enabling the application of the method in situations where time is a scarce resource to the detriment of a percentage of performance.

Table 2. Performance in percentage of the AO with respect to exhaustive search.

Dataset	Objective function	Time % (seconds)	Performance %
UCI Heart Disease	G-Mean	32.13%	100%
	Ponderated	35.99%	100%
	Accuracy	34.27%	100%
Diabetes	G-Mean	11.01%	99.34%
	Ponderated	12.07%	100.13%
	Accuracy	14.48%	100.37%
UCI Cardiotocography 3-Class	G-Mean	9.00%	99.54%
	Ponderated	9.65%	99.55%
	Accuracy	11.58%	100.11%

5 Conclusions

In this paper, an adaptation of the AO algorithm applied to the metaheuristic optimization of the hyperparameter $EB_{Nº Neural}$ of the ELM, was introduced. The results demonstrate that the behavior of the aquilas inspires the algorithm. At the same time, they hunt their prey to survive, allowing the optimization of objective functions in a reduced time, while its value is near to the optimal global compared to an Exhaustive Search approach.

Although the ELM used in this research can be considered a low-complexity model compared to other neural networks with more hyperparameters, its true potential lies in its training speed and high generalization ability despite associated randomness. For more complex ELM models, the hyperparameter search space can be so extensive that traditional optimization methods may not be feasible. The AO algorithm shows excellent potential in optimizing processing times. Future research should consider this potential. Compared to more complex ELM variations, such as Regularized ELM or Multi-layer ELM, which have a higher number of hyperparameters, using the AO algorithm may significantly reduce the number of trains without significantly affecting performance. This means there will be greater availability of hardware resources and energy savings, which should not be underestimated.

It's clear from the graphs that maximizing the Accuracy metric in unbalanced multiclass problems can negatively impact the G-Mean. This underscores the need for new objective functions to optimize, such as the average of both. By doing so, we can avoid the situation where improving one metric comes at the cost of reducing other important metrics, potentially leading to overfitting and other model problems.

Applying soft computing algorithms, particularly the AO metaheuristic, to this problem is highly advantageous, especially when time is critical. The AO metaheuristic, with its proven robustness, low computational cost, and linear algorithmic complexity (O(n)), instills confidence in its ability to promptly deliver close to global best optimal solutions.

References

1. Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A.A., Al-Qaness, M.A., Gandomi, A.H.: Aquila optimizer: a novel meta-heuristic optimization algorithm. Computers & Industrial Engineering **157**, 107250 (2021)
2. Curry, H.B.: The method of steepest descent for non-linear minimization problems. Quarterly of Applied Mathematics **2**(3), 258–261 (1944)
3. Gonzalez, P., Iglesias, P., Silva, E.: Restricted particle swarm optimization meta-heuristic method. In: 2023 42nd IEEE International Conference of the Chilean Computer Science Society (SCCC). pp. 1–5 (2023). https://doi.org/10.1109/SCCC59417.2023.10315753
4. González-Gutiérrez, P., Vásquez, P., Alcaino-Jaque, B.E., Barrientos, R., Mora, M., Tirado-Marabolí, F., Tauber, C.: Heuristic parametrization of anisotropic diffusion filtering. Proceedings - International Conference of the Chilean Computer Science Society, SCCC **2018-November** (2019)
5. Holland, J.H.: Genetic algorithms. Scientific american **267**(1), 66–73 (1992)
6. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: Theory and applications. Neurocomputing **70**(1), 489–501 (2006). https://doi.org/https://doi.org/10.1016/j.neucom.2005.12.126, https://www.sciencedirect.com/science/article/pii/S0925231206000385, neural Networks
7. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Tech. rep., Technical report-tr06, Erciyes university, engineering faculty, computer engineering department (2005)
8. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks. vol. 4, pp. 1942–1948. IEEE (1995)
9. Pao, Y.H., Park, G.H., Sobajic, D.J.: Learning and generalization characteristics of the random vector functional-link net. Neurocomputing **6**(2), 163–180 (1994). https://doi.org/https://doi.org/10.1016/0925-2312(94)90053-1, https://www.sciencedirect.com/science/article/pii/0925231294900531, backpropagation, Part IV
10. Paz, K.: Media aritmética simple. Facultad de Ingeniería **7**, 1–13 (2007)
11. Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. IEEE Transactions on pattern analysis and machine intelligence **12**(7), 629–639 (1990)
12. Schmidt, W., Kraaijveld, M., Duin, R.: Feedforward neural networks with random weights. In: Proceedings., 11th IAPR International Conference on Pattern Recognition. Vol.II. Conference B: Pattern Recognition Methodology and Systems. pp. 1–4 (1992). https://doi.org/10.1109/ICPR.1992.201708