ACCSCIENCE PUBLISHING
An International Journal of Optimization and Control: Theories & Applications
ISSN: 2146-0957 eISSN: 2146-5703
Vol.16, No.4, pp.1216-1232 (2026)
https://doi.org/10.36922/IJOCTA026150058
IJOCTA
RESEARCH ARTICLE

A snake optimizer-based kernel extreme learning machine for classification

Haolong Yang[1], Qi Diao[1,2*] and WengHowe Chan[2*]

[1]Faculty of Artificial Intelligence, Zhejiang Dongfang Polytechnic, Wenzhou, China
[2]Faculty of Computing, Universiti Teknologi Malaysia, Skudai, Johor, Malaysia
yanghaolong@wzut.edu.cn; diaoqi@graduate.utm.my; cwenghowe@utm.my

ARTICLE INFO

Article history:
Received: April 7, 2026
Revised: May 12, 2026
Accepted: May 19, 2026
Published Online: June 5, 2026

Keywords:
Extreme learning machine
Snake optimizer
Swarm intelligence
Parameter optimization
Classification

AMS Classification 2010:
26A33; 34A08; 35H15; 34K50
47H10; 60H10

ABSTRACT

Kernel Extreme Learning Machine (KELM) is well-suited for nonlinear classification, but its performance is highly sensitive to the regularization parameter C and the RBF kernel width σ. To reduce manual tuning, we proposes a Snake Optimizer-based KELM model, termed SKELM, which jointly optimizes C and σ using validation accuracy as the fitness function. Experiments on ten UCI benchmark datasets indicate that compared with KELM, LSOKELM, and SSAKELM, SKELM achieves the unique best accuracy on five datasets and ties for the best accuracy on the remaining three datasets. The proposed method also shows fewer average iterations and a lower standard deviation under the current experimental setting. These results suggest that SKELM can serve as a effective parameter-search framework for KELM, although further validation with additional baselines, statistical tests, convergence curves, runtime comparisons, and more evaluation metrics remains necessary.

1. Introduction

Pattern classification is a fundamental task in machine learning and artificial intelligence, with applications in image recognition, fault diagnosis, medical detection, financial risk control, environmental monitoring, and other fields.[1,2] Its main objective is to construct accurate and efficient models for identifying the class labels of unknown samples.[3] With the rapid growth in the volume and complexity of data, classification tasks face increasing challenges from high dimensionality, nonlinear feature relationships, and complex sample distributions.[4,5] Conventional classification methods, such as support vector machines and traditional neural networks, may be limited by slow training, limited generalization, or difficult y in parameter tuning in some practical scenarios.[6,7]

Recent studies have demonstrated the effectiveness of machine learning and intelligent models in complex engineering and decision-making applications. For example, Kayıran applied machine learning techniques to the thermo-elastic analysis of fiber-reinforced composite rotating disks, showing that data-driven models can assist complex engineering analysis. Musbah and Badi developed an LLM-assisted virtual expert weight elicitation framework based on Z-number multi-agent decision modeling for pharmaceutical supply chains, indicating the growing role of intelligent learning and decision-support methods in uncertain practical environments.[8,9] These studies suggest that machine learning and intelligent optimization methods are increasingly used not only for prediction tasks, but also for parameter analysis, complex modeling, and decision-support problems.For clarity, the main abbreviations used in this paper are summarized in **Appendix A**.

Extreme Learning Machine (ELM), as a single-hidden-layer feedforward neural network, has attracted considerable attention in classification

*Corresponding Author

and regression tasks due to its simple structure and fast training speed.[10–12] Unlike traditional neural networks that rely on iterative backpropagation, ELM reduces computational cost by randomly assigning the hidden-layer parameters and then analytically determines the output weights.[13] However, the randomness of hidden-layer parameters may lead to unstable performance and weak generalization in practical applications.[14,15] This limitation becomes more evident in small-sample or noisy datasets, where stable generalization is important.

To overcome this limitation, Kernel Extreme Learning Machine (KELM) introduces kernel functions into the ELM framework to implicitly map input samples into a high-dimensional feature space.[16] By using the kernel trick, KELM avoids the explicit construction of hidden-layer nodes and improves the nonlinear modeling capability and stability of ELM.[17] KELM has been successfully applied to various classification problems.[18,19] Nevertheless, its performance is highly sensitive to the regularization parameter C and the RBF kernel width σ. Improper parameter settings may lead to overfitting, underfitting, or unstable classification results, which limits further improvement of KELM.[20]

Simultaneously,recent neural network studies have increasingly emphasized not only prediction accuracy but also stability, bifurcation behavior, time-delay effects, and control mechanisms. Maharajan *et al.*[21] investigated the Lagrange stability of inertial-type neural networks using a Lyapunov-Krasovskii functional approach. Cui *et al.*[22] studied bifurcation and controller design for five-dimensional bidirectional associative memory neural networks with time delay. Xu *et al.* [23] further explored bifurcation problems in fractional-order octonion-valued neural networks involving delays, while Xu *et al.*[24] investigated bifurcation and control schemes for fractional neural networks with multiple delays. These studies show that stability, parameter sensitivity, and dynamic robustness have become important concerns in neural network research. Although KELM is not a recurrent or delayed neural network model, its classification performance is also strongly affected by parameter selection. Therefore, improving the parameter adaptability of KELM is consistent with the broader research trend of enhancing the reliability of neural network-based models.

Swarm intelligence optimization algorithms have been widely used to solve parameter optimization problems in machine learning models[25,26] Representative algorithms, such as Particle Swarm Optimization (PSO), [27] Genetic Algorithm (GA),[28] Sparrow Search Algorithm (SSA), [29] and Lion Swarm Optimization (LSO),[30] have been introduced to optimize KELM parameters, leading to improved classification performance. [31,32] However, different swarm intelligence-based KELM optimization methods may vary in their search mechanisms, convergence behavior, and ability to avoid local optima.

In addition to swarm intelligence optimization, recent studies on intelligent decision-making and uncertainty modeling also provide useful insights for parameter selection and model reliability. Senapati *et al.* proposed an artificial intelligence-driven group decision-making method using Sugeno-Weber triangular norms in a dual hesitant q-rung orthopair fuzzy context for healthcare supply chain management. [33] Khan *et al.* developed a complex interval-valued intuitionistic fuzzy decision support system for COVID-19 healthcare facility evaluation. [34] Kakati *et al.* introduced rectified complex T-spherical fuzzy Dombi-Choquet integral operators and applied them to diabetic retinopathy detection using fundus images. [35] Although these studies are not KELM-based classifiers, they show that uncertainty-aware decision mechanisms and intelligent optimization strategies are valuable for improving model reliability in complex application scenarios.

Snake Optimizer (SO) is a recently developed swarm intelligence algorithm inspired by snake foraging, fighting, mating, and reproduction behaviors in nature. [36] By incorporating food quantity and temperature control mechanisms, SO dynamically balances global exploration and local exploitation during the optimization process. This feature makes SO potentially suitable for nonlinear and multimodal parameter optimization problems. However, the application of SO to the parameter optimization of kernel-based learning models has not been sufficiently investigated, especially for the coupled optimization of the regularization parameter C and the RBF kernel width σ in KELM.

Based on the above analysis, the research gap of this study can be summarized as follows. Although several metaheuristic algorithms have been used for KELM parameter optimization, most existing studies mainly focus on improving classification accuracy by searching for suitable values of C and σ. Less emphasis on how different optimizer mechanisms affect the coupled parameter-search process, especially in terms of search stability, convergence behavior, and robustness under the same experimental configuration. Moreover, the application of SO to KELM

parameter optimization remains relatively limited. Therefore, this study raises the following research question: can the Snake Optimizer provide a useful alternative parameter-search mechanism for KELM by jointly optimizing the regularization parameter C and the RBF kernel width σ?

To address this question,we propose a Snake Optimizer-based Kernel Extreme Learning Machine, termed SKELM, for classification tasks. In the proposed framework, each snake individual represents a candidate parameter pair (C, σ), and classification accuracy on the validation set is adopted as the fitness function. The main objective is to reduce manual parameter tuning and examine whether SO can provide a useful parameter-search strategy for KELM under controlled experimental settings.

The main contributions of this paper are summarized as follows:

- A Snake Optimizer-driven parameter tuning framework is developed for KELM to jointly optimize the regularization parameter C and the RBF kernel width σ.
- A compact two-parameter encoding strategy and an accuracy-based fitness evaluation mechanism are designed to support the coupled parameter search of KELM.
- Experiments on ten UCI benchmark datasets are conducted to provide preliminary evidence of the effectiveness of SKELM under controlled parameter settings. The results show that SKELM achieves competitive classification accuracy compared with KELM, LSOKELM, and SSAKELM, while its broader superiority still requires further validation with additional baselines and statistical tests.

2. Kernel extreme learning machine

Extreme Learning Machine (ELM) is a single-hidden-layer feedforward neural network in which the input weights and hidden-layer biases are randomly generated and remain fixed during training. [37,38] Given the hidden-layer output matrix, the output weights can be analytically obtained by solving a least-squares problem, which enables fast training. However, the random assignment of hidden-layer parameters may introduce performance fluctuations and weaken the generalization ability of ELM in practical applications. [17]

To overcome this limitation, Kernel Extreme Learning Machine (KELM) incorporates kernel functions into the ELM framework. [38] By implicitly mapping input samples into a high-dimensional feature space, KELM avoids the explicit construction of hidden-layer nodes and improves the nonlinear modeling capability of ELM. Owing to the kernel trick, KELM can reduce the sensitivity to randomly generated hidden-layer parameters and has been widely applied to classification tasks.

The main merit of KELM lies in its ability to combine the fast analytical learning property of ELM with the nonlinear representation capability of kernel methods. Compared with conventional ELM, KELM replaces the explicit hidden-layer mapping with a kernel matrix. Unlike traditional iterative neural networks, KELM avoids iterative backpropagation for output-weight learning and obtains the output weights analytically. Moreover, by using kernel functions such as the radial basis function (RBF) kernel, KELM can handle nonlinear classification problems without explicitly constructing high-dimensional feature mappings.

Consider a training dataset

$$\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{t}_i) \mid \boldsymbol{x}_i \in \mathbb{R}^d,\ \boldsymbol{t}_i \in \mathbb{R}^c,\ i = 1, 2, \ldots, N\},$$

where $\boldsymbol{x}_i$ denotes the input sample, $\boldsymbol{t}_i$ denotes the corresponding target output, d denotes the input dimension, c denotes he number of output classes, and N denotes the number of training samples. Let $\boldsymbol{H}$ denote the hidden-layer output matrix of ELM. The output weight matrix $\boldsymbol{\beta}$ can be obtained by minimizing the following regularized least-squares objective function:

$$\min_{\boldsymbol{\beta}} \|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{T}\|_F^2 + \frac{1}{C}\|\boldsymbol{\beta}\|_F^2, \tag{1}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, C denotes the regularization parameter, and $\boldsymbol{T}$ denotes the target output matrix. The first term measures the empirical fitting error between the network output and the target matrix, while the second term controls the magnitude of the output weights.

The analytical solution of $\boldsymbol{\beta}$ can be expressed as

$$\boldsymbol{\beta} = \boldsymbol{H}^{\mathrm{T}} \left(\frac{1}{C}\boldsymbol{I} + \boldsymbol{H}\boldsymbol{H}^{\mathrm{T}} \right)^{-1} \boldsymbol{T}. \tag{2}$$

By introducing a kernel function $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$, the kernel matrix of KELM is defined as

$$\boldsymbol{\Omega}(i, j) = K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^{\mathrm{T}} \phi(\boldsymbol{x}_j), \tag{3}$$

where $\phi(\cdot)$ denotes the implicit nonlinear mapping induced by the kernel function.

In this study, the radial basis function (RBF) kernel is adopted:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{2\sigma^2}\right), \quad (4)$$

where σ denotes the RBF kernel width.

Accordingly, the output function of KELM for a test sample $\boldsymbol{x}$ can be written as

$$f(\boldsymbol{x}) = \boldsymbol{k}(\boldsymbol{x})\left(\frac{1}{C}\boldsymbol{I} + \boldsymbol{\Omega}\right)^{-1}\boldsymbol{T}, \quad (5)$$

where

$$\boldsymbol{k}(\boldsymbol{x}) = [K(\boldsymbol{x}, \boldsymbol{x}_1), K(\boldsymbol{x}, \boldsymbol{x}_2), \ldots, K(\boldsymbol{x}, \boldsymbol{x}_N)].$$

Although KELM has nonlinear learning capability and avoids the explicit random hidden-layer mapping used in standard ELM, its classification performance remains highly sensitive to the selection of the regularization parameter C and the RBF kernel width σ. Therefore, swarm intelligence algorithms have been increasingly used to search for suitable values of these two parameters, aiming to improve the parameter adaptability and classification performance of KELM.

3. Snake optimizer

Snake Optimizer (SO) is a swarm intelligence optimization algorithm inspired by the foraging, fighting, mating, and reproduction behaviors of snakes in nature. [36] Similar to other population-based metaheuristic algorithms, SO searches for a solution by iteratively updating a population of candidate solutions. Its search behavior is mainly controlled by food quantity and environmental temperature factors, which are used to switch among different behavioral phases. These mechanisms enable SO to balance global exploration and local exploitation during the optimization process.

In machine learning parameter optimization, metaheuristic algorithms are often used to handle nonlinear, multimodal, and non-convex search spaces where gradient information is unavailable or difficult to obtain. For KELM, the regularization parameter C and the RBF kernel width σ form a coupled continuous optimization problem. Improper parameter settings may lead to overfitting, underfitting, or unstable classification performance. Therefore, SO is introduced in this study as an alternative search strategy for selecting suitable KELM parameters.

3.1. Algorithm principle

In SO, each snake individual represents a candidate solution in the search space. The position of the i-th snake at iteration t is defined as

$$\boldsymbol{X}_i^t = [x_{i1}^t, x_{i2}^t, \ldots, x_{id}^t], \quad (6)$$

where d denotes the dimensionality of the optimization problem. In the proposed SKELM framework, the search dimension is $d = 2$, and each individual corresponds to a candidate KELM parameter pair (C, σ).

SO introduces two control factors: the food quantity factor $Q(t)$ and the environmental temperature factor $\mathcal{T}(t)$. They are defined as

$$Q(t) = c_1 \exp\left(\frac{t - t_{\max}}{t_{\max}}\right), \quad (7)$$

and

$$\mathcal{T}(t) = \exp\left(-\frac{t}{t_{\max}}\right), \quad (8)$$

where c_1 is a constant coefficient and $t_{\max}$ is the maximum number of iterations.

The values of $Q(t)$ and $\mathcal{T}(t)$ determine the behavioral phase of the snake population. When the food quantity is insufficient, namely $Q(t) < 0.25$, the population tends to perform exploration by moving individuals toward randomly selected positions or individuals. When the food quantity is sufficient, namely $Q(t) \geq 0.25$, the population enters the exploitation stage. If the environmental temperature is high, for example $\mathcal{T}(t) > 0.6$, snakes move toward the food position, which corresponds to the current best solution. If the temperature is relatively low, the population switches between fighting and mating behaviors according to a random probability. These mechanisms allow SO to adjust its search behavior during different stages of optimization.

3.2. Implementation details of SO in SKELM

In the proposed SKELM model, SO is used for the two-dimensional hyperparameter search of KELM. Each candidate solution is encoded as

$$\boldsymbol{X}_i^t = [C_i^t, \sigma_i^t], \quad (9)$$

where C_i^t and σ_i^t denote the regularization parameter and the RBF kernel width of the i-th individual at iteration t, respectively.

The population is initialized randomly within the predefined search space:

$$X_{ij}^0 = L_j + r_{ij}(U_j - L_j), \quad j = 1, 2, \quad (10)$$

where L_j and U_j are the lower and upper bounds of the j-th parameter, and r_{ij} is a random number uniformly distributed in $[0, 1]$. In this study, the two dimensions correspond to C and σ, respectively.

During the optimization process, SO updates the

candidate solutions according to its behavioral rules, including exploration, movement toward food, fighting, mating, and reproduction. In the reproduction phase, if the egg-hatching condition is satisfied, a newly generated individual is used to replace the worst individual in the current population. This replacement mechanism helps maintain population diversity and reduces the chance of being trapped in an unfavorable search region.

For each updated individual, boundary control is applied as follows:

$$X_{ij}^{t} = \begin{cases} L_j, & X_{ij}^{t} < L_j, \\ U_j, & X_{ij}^{t} > U_j, \\ X_{ij}^{t}, & \text{otherwise.} \end{cases} \tag{11}$$

This operation ensures that all candidate parameter pairs remain within the valid search ranges during the optimization process.

In each iteration, the fitness value of a candidate solution is calculated by training a KELM model with the corresponding parameter pair (C_i, σ_i) on the training subset and evaluating its classification accuracy on the validation subset:

$$\text{Fitness}(\boldsymbol{X}_i) = \text{Acc}_{val}(C_i, \sigma_i). \tag{12}$$

Since this study aims to maximize validation accuracy, the candidate solution with the highest fitness value is retained as the current best solution. After the maximum number of iterations is reached, the best parameter pair (C^*, σ^*) is used to train the final KELM classifier.

3.3. Algorithm procedure

The general procedure of SO for KELM parameter optimization can be summarized as follows:

(1) Initialization: Randomly initialize the snake population within the predefined search range and divide the population into male and female groups according to the SO mechanism.
(2) Fitness Evaluation: Evaluate the fitness value of each individual. In this study, the fitness value is determined by the validation classification accuracy of KELM under the corresponding parameter pair (C, σ).
(3) Behavior Determination: Compute the food quantity factor $Q(t)$ and the environmental temperature factor $\mathcal{T}(t)$ to determine the current behavioral phase.
(4) Position Update: Update the positions of snake individuals according to the corresponding SO behavioral rules, including exploration, movement toward food, fighting, mating, and reproduction.
(5) Boundary Control: Ensure that each updated candidate solution remains within the predefined search ranges of C and σ.
(6) Best Solution Update: Re-evaluate the updated individuals and update the current global best solution if a better parameter pair is found.
(7) Termination: Repeat the above steps until the maximum number of iterations is reached, and output the best parameter pair as the optimized KELM configuration.

4. Proposed SKELM model

To address the parameter sensitivity and manual tuning limitations of Kernel Extreme Learning Machine (KELM), this paper proposes a Snake Optimizer-based KELM classification framework, termed SKELM. In this framework, the Snake Optimizer (SO) is served as a global metaheuristic search strategy to identify a suitable combination of the regularization parameter C and the RBF kernel width σ. By optimizing these two coupled parameters, SKELM aims to reduce manual parameter tuning and examine whether the behavioral search mechanism of SO can provide a useful alternative for KELM parameter selection. [39]

Compared with existing swarm intelligence-based KELM optimization methods, the proposed SKELM has several potential merits. First, SO uses food quantity and environmental temperature factors to adaptively control the transition between exploration and exploitation, allowing the optimizer to search broadly in the early stage and refine promising regions in the later stage. Second, the fighting, mating, and reproduction mechanisms of SO help maintain population diversity during the search process. Third, SKELM adopts a compact two-dimensional encoding strategy, in which each individual directly represents a candidate parameter pair (C, σ). This design makes the optimization process simple, interpretable, and suitable for the coupled parameter search problem of KELM.

Within the proposed SKELM framework, each snake individual represents a candidate hyperparameter configuration of the KELM classifier. The i-th individual is formulated as

$$\boldsymbol{X}_i = [C_i, \sigma_i], \tag{13}$$

where C_i denotes the regularization parameter and σ_i denotes the RBF kernel width. The corresponding search space is defined as

$$C \in [C_{\min}, C_{\max}], \quad \sigma \in [\sigma_{\min}, \sigma_{\max}]. \tag{14}$$

In this study, the search ranges of the two opti-

mized KELM parameters are set as

$$C \in [10^{-3}, 10^{3}], \quad \sigma \in [10^{-3}, 10^{3}]. \tag{15}$$

The same search ranges are used for all metaheuristic-based KELM methods.

During the optimization process, SO iteratively updates the population according to its phase-driven behavioral rules, including exploration, exploitation, fighting, mating, and reproduction. Boundary control is applied after each position update to ensure that all candidate solutions remain within the predefined search ranges. At each iteration, the fitness of an individual is evaluated by training a KELM classifier with the corresponding parameter pair and measuring its classification accuracy on the validation set. The fitness function is defined as

$$\text{Fitness}(\boldsymbol{X}_i) = \text{Acc}_{val}(\boldsymbol{X}_i), \tag{16}$$

where $\text{Acc}_{val}(\boldsymbol{X}_i)$ denotes the validation accuracy obtained by the KELM model parameterized by $\boldsymbol{X}_i$.

Through iterative fitness evaluation, population evolution, boundary control, and global best solution tracking, SKELM searches for a suitable parameter pair (C^*, σ^*). After the termination criterion is satisfied, the final KELM classifier is retrained using (C^*, σ^*) on the combined training and validation datasets. The generalization performance of the resulting model is then evaluated on an independent test set.

The overall workflow of the proposed SKELM algorithm is illustrated in **Figure 1**, and the detailed procedural steps are summarized in **Table 1**.

As shown in **Table 1**, the proposed SKELM algorithm follows an optimization-training proce

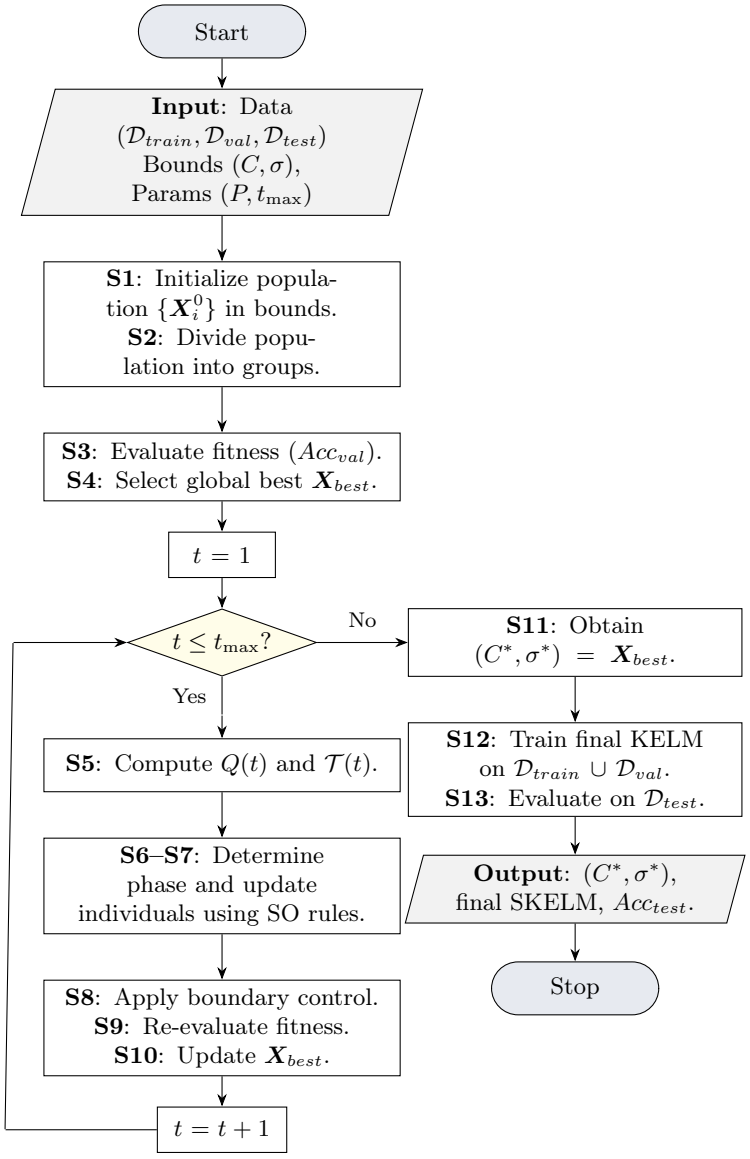

Figure 1. Flowchart of the proposed SKELM algorithm

Table 1. Procedure of the proposed SKELM classification algorithm

Step	Description
Input	Training set $\mathcal{D}_{train}$, validation set $\mathcal{D}_{val}$, and test set $\mathcal{D}_{test}$; search bounds $C \in [C_{\min}, C_{\max}]$ and $\sigma \in [\sigma_{\min}, \sigma_{\max}]$; population size P and maximum iterations $t_{\max}$.
Output	Optimized parameters (C^*, σ^*), trained SKELM classifier, and test accuracy Acc_{test}.
Step 1	Initialize a population $\{\boldsymbol{X}_i^0\}_{i=1}^P$, where each individual $\boldsymbol{X}_i^0 = [C_i^0, \sigma_i^0]$ is randomly generated within the predefined bounds.
Step 2	Divide the population into groups according to the SO mechanism.
Step 3	Evaluate the fitness of each individual by training a KELM classifier with (C_i^0, σ_i^0) on $\mathcal{D}_{train}$ and computing the validation accuracy Acc_{val} on $\mathcal{D}_{val}$.
Step 4	Select the best individual as the initial global best solution $\boldsymbol{X}_{best}$.
Step 5	For iteration $t = 1$ to $t_{\max}$, compute the food quantity factor $Q(t)$ and the environmental temperature factor $\mathcal{T}(t)$.
Step 6	Determine the behavioral phase according to $Q(t)$ and $\mathcal{T}(t)$.
Step 7	Update each individual $\boldsymbol{X}_i^t$ according to the corresponding SO behavioral rules.
Step 8	Apply boundary control to ensure that all candidate solutions remain within the predefined search ranges.
Step 9	Re-evaluate the fitness of each updated individual.
Step 10	Update the global best solution $\boldsymbol{X}_{best}$ if a better parameter pair is found.
Step 11	Obtain the optimized parameters $(C^*, \sigma^*) = \boldsymbol{X}_{best}$.
Step 12	Train the final KELM classifier using (C^*, σ^*) on $\mathcal{D}_{train} \cup \mathcal{D}_{val}$.
Step 13	Evaluate the final model on $\mathcal{D}_{test}$ to obtain Acc_{test}.

-dure. First, each snake individual is encoded as a candidate parameter pair (C, σ). Then, SO iteratively updates the population according to its behavioral rules and evaluates each candidate solution using validation accuracy. After the maximum number of iterations is reached, the best parameter pair (C^*, σ^*) is selected to train the final KELM classifier, which is then evaluated on the independent test set. This procedure separates parameter optimization from final model evaluation.

5. Experimental results and analysis

5.1. Experimental setup

To comprehensively evaluate the classification performance, convergence behavior, and stability of the proposed SKELM, experiments were conducted on ten benchmark datasets from the UCI repository,[40] including Wine, Balance, Zoo, Heart, WDBC, Iris, Soybean, P_gene, Glass, and Haberman. These datasets cover different sample sizes, feature dimensions, and classification characteristics, thereby providing a representative basis for evaluating the robustness and generalization ability of the proposed method. **Table 2** summarizes the basic information of the ten UCI benchmark datasets used in the experiments, including the number of samples, features, classes, and the adopted training ratios.

All experimental results are reported as the average performance over 20 independent runs with different random data partitions and optimizer initializations.

All experiments were implemented in MATLAB R2017a under the same computational environment to ensure a fair comparison among different methods. For each dataset, the samples were randomly divided into training and testing subsets according to the specified training ratio. During the parameter optimization stage, a validation subset was further separated from the training data to calculate the validation accuracy Acc_{val}, which was used as the fitness value for the optimization algorithm. After the optimal parameter pair (C^*, σ^*) was obtained, the final KELM classifier was retrained using the combined training and validation data and then evaluated on the independent testing set.

To ensure a controlled comparison, the same random data partitions were used for all compared methods in each independent run. In other words, KELM, LSOKELM, SSAKELM, and SKELM were evaluated on the same training, validation, and testing subsets under the corresponding run. For the metaheuristic-based methods, the same population size, maximum number of iterations, search dimension, and parameter search ranges were adopted. Therefore, the reported differences mainly reflect the behavior of the optimization

strategies under the current experimental configuration rather than differences in data splitting or search-space settings.

5.2. Kernel function and parameter settings

For all KELM-based methods, the radial basis function (RBF) kernel was adopted to ensure a fair and consistent comparison. The RBF kernel is widely used for nonlinear classification tasks because it can implicitly map input samples into a high-dimensional feature space and capture complex nonlinear relationships among samples. In the proposed SKELM and other metaheuristic-based KELM variants, the regularization parameter C and the RBF kernel width σ were optimized simultaneously.

For all optimization algorithms, the maximum number of iterations was fixed at 100, the population size was set to 30, and the search dimension was set to 2, corresponding to the two optimized KELM parameters (C, σ). The same optimization configuration was used for LSOKELM, SSAKELM, and SKELM. Therefore, the observed performance differences mainly reflect the search capability and optimization mechanism of each algorithm rather than differences in experimental settings.

5.3. Classification performance comparison

When the random proportion of training data in the Wine dataset is set to 0.5, the classification accuracies of KELM, LSOKELM, SSAKELM, and SKELM are 91.11%, 94.62%, 94.22%, and 98.01%, respectively, as shown in **Figure 2**. Compared with the original KELM, the proposed SKELM improves the classification accuracy by 6.90 percentage points. It also outperforms LSOKELM and SSAKELM by 3.39 and 3.79 percentage points, respectively. This result suggests that the Snake Optimizer may provide a useful parameter search strategy for KELM on the Wine dataset under the current experimental setting.

When the random proportion of training data in the Balance dataset is set to 0.5, the classification accuracies of KELM, LSOKELM, SSAKELM, and SKELM are 97.34%, 97.76%, 97.12%, and 99.36%, respectively. The classification comparison is shown in **Figure 3**. Although all methods achieve relatively high classification accuracy on this dataset, SKELM still achieves the best result. This result indicates that the proposed method still yields a high classification accuracy on this dataset under the adopted experimental setting.

Future work will provide dataset-level standard deviations, confidence intervals, convergence curves, and runtime comparisons to support a more rigorous stability analysis.

As shown in **Table 3**, the proposed SKELM achieves the unique best classification accuracy on five datasets, namely Zoo, Heart, WDBC, P_gene, and Haberman, and obtains the tied best performance on three datasets, namely Iris, Soybean, and Glass. These results indicate that SKELM achieves competitive classification per-

Table 2. Summary of the UCI benchmark datasets used in the experiments

Dataset	Samples	Features	Classes	Training Ratio
Wine	178	13	3	0.5
Balance	625	4	3	0.5
Zoo	101	16	7	0.3
Heart	270	13	2	0.3
WDBC	569	30	2	0.4
Iris	150	4	3	0.4
Soybean	307	35	19	0.4
P_gene	106	57	2	0.5
Glass	214	9	6	0.5
Haberman	306	3	2	0.5

Notes: Soybean refers to the Soybean Large dataset, which contains 307 instances, 35 attributes, and 19 disease classes. [41] P_gene refers to the Promoter Gene Sequences dataset, which contains 106 DNA sequences with 57 nucleotide-position features for binary promoter classification. [42]

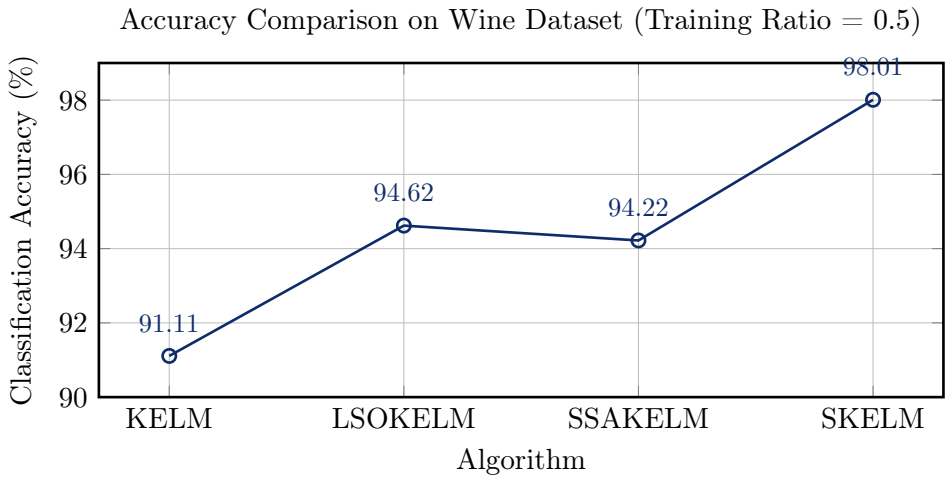

Figure 2. Comparison of classification accuracy of different algorithms on the Wine dataset with a training ratio of 0.5.

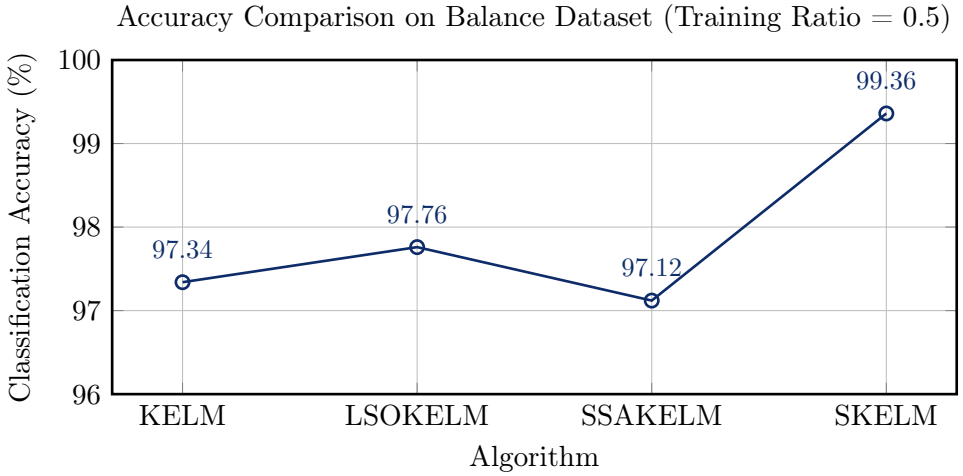

Figure 3. Comparison of classification accuracy of different algorithms on the Balance dataset with a training ratio of 0.5.

formance across the tested datasets under the current experimental setting. However, because the comparison is limited to KELM, LSOKELM, and SSAKELM, these results should be interpreted as preliminary evidence rather than conclusive proof of general superiority over all parameter optimization strategies.

Compared with the original KELM, SKELM improves the classification accuracy on most datasets. This improvement is consistent with the observation that the performance of KELM is strongly affected by the selection of the regularization parameter C and the RBF kernel width σ. When these parameters are not properly selected, KELM may suffer from underfitting, overfitting, or unstable decision boundaries. By using the Snake Optimizer to search for a suitable parameter pair (C, σ), SKELM reduces the dependence on manual parameter tuning and improves the adaptability of KELM to datasets with different data distributions.

Compared with LSOKELM and SSAKELM, SKELM yields competitive performance on several datasets. The improvement is particularly evident on P_gene and Haberman. On the P_gene dataset, SKELM achieves 83.33%, outperforming KELM, LSOKELM, and SSAKELM by 20.37, 1.85, and 7.40 percentage points, respectively. On the Haberman dataset, SKELM reaches 79.87%, which is also higher than all compared methods. These results suggest that the food-quantity and environmental-temperature mechanisms of SO can provide a superior balance between global exploration and local exploitation when searching in the coupled parameter space of KELM.

It should be noted that several methods achieve 100.00% accuracy on Iris, Soybean, and Glass. This indicates that these datasets are relatively easier for the compared KELM-based classifiers

under the given training ratios. Therefore, the results on these datasets provide limited evidence for distinguishing the optimization capabilities of different algorithms. In contrast, the performance gains on more challenging datasets, such as P_gene, Haberman, Heart, and WDBC, provides stronger evidence for the effectiveness of SKELM.

Overall, the classification results suggest that SKELM can provide a feasible SO-based parameter optimization mechanism for KELM. Nevertheless, additional comparisons with grid-search KELM, random-search KELM, PSO-KELM, GA-KELM, and RBF-SVM would be needed in future work to further verify the relative advantage of the proposed method.

5.4. Discussion on convergence speed and stability

In addition to classification accuracy, convergence behavior and result variability are also useful references for evaluating metaheuristic-based KELM optimization methods. In this study, the average number of iterations and the standard deviation of accuracy over repeated runs were recorded as descriptive indicators. It should be noted that these indicators provide only preliminary evidence of convergence and stability, since detailed convergence curves, runtime comparisons, and statistical significance tests were not included in the present study.

Table 4 presents the descriptive convergence and variability results of the compared methods. As discussed above, these values should be regarded as preliminary indicators rather than definitive
evidence of convergence superiority or robustness.

Among the metaheuristic-based methods, SKELM requires 35 average iterations, while LSOKELM and SSAKELM require 42 and 68 average iterations, respectively. This observation suggests that SO may locate a suitable parameter pair within fewer iterations in the reported experiments. However, since no complete convergence curves or runtime comparisons are provided, this result should be interpreted cautiously as a descriptive comparison rather than a definitive conclusion.

In terms of result variability, SKELM obtains a standard deviation of 0.96%, which is lower than those of KELM, LSOKELM, and SSAKELM in the current experimental records. This indicates that SKELM produced relatively consistent accuracy values across the repeated runs. Nevertheless, the robustness claim remains preliminary since standard deviations for each individual dataset and statistical significance tests are not reported. Future work will provide dataset-level standard deviations, confidence intervals, convergence curves, and runtime comparisons to support a more rigorous stability analysis.

Among the metaheuristic-based methods, SKELM achieves the fastest convergence, requiring only 35 average iterations to reach the optimized solution. This is fewer than LSOKELM with 42 iterations and much fewer than SSAKELM with 68 iterations. The result suggests that SO can search the coupled parameter space of the regularization parameter C and the RBF kernel width σ more efficiently. The adaptive food-quantity and environmental-temperature mechanisms allow the optimizer to conduct global exploration in the early stage and strengthen local exploitation in promising regions during the later stage, thereby accelerating the convergence process.

In terms of stability, SKELM obtains the lowest standard deviation of 0.96%, outperforming KELM, LSOKELM, and SSAKELM. A lower standard deviation indicates that SKELM produces more consistent results across repeated

Table 3. Classification accuracy (%) of different algorithms on UCI datasets

Dataset (Training Ratio)	KELM	LSOKELM	SSAKELM	SKELM
zoo (0.3)	90.41	91.78	90.41	**93.15**
heart (0.3)	55.03	61.90	62.43	**62.96**
wdbc (0.4)	88.05	90.09	88.34	**91.55**
iris (0.4)	100.00	100.00	100.00	100.00
soybean (0.4)	100.00	100.00	100.00	100.00
p_gene (0.5)	62.96	81.48	75.93	**83.33**
glass (0.5)	100.00	85.32	100.00	100.00
haberman (0.5)	74.03	75.97	74.03	**79.87**

Table 4. Descriptive convergence and stability comparison of different algorithms

Algorithm	Avg. iterations	Std. Dev. (%)	Remark
KELM	–	1.92	No optimization
LSOKELM	42	1.15	Optimization-based
SSAKELM	68	1.48	Optimization-based
SKELM	**35**	**0.96**	Optimization-based

independent runs. This improvement can be attributed to the population diversity and adaptive behavioral update mechanisms of SO, which reduce the risk of premature convergence and enhance robustness during parameter search.

Overall, the convergence and stability results further support the effectiveness of the proposed SKELM model. Compared with existing KELM-based optimization methods, SKELM not only improves classification accuracy but also achieves a better balance among optimization efficiency, convergence stability, and robustness.

5.5. Computational complexity discussion

The computational cost of SKELM mainly consists of KELM training, validation prediction, and SO-based repeated candidate evaluation. For one KELM training process, constructing the RBF kernel matrix requires pairwise similarity computation among training samples, with a complexity of approximately $O(N^2d)$, where N is the number of training samples and d is the feature dimension. Solving the regularized linear system generally requires $O(N^3)$ operations.

For each candidate parameter pair (C, σ), the KELM model must be trained on the training subset and evaluated on the validation subset. If the validation set contains N_v samples, the validation prediction cost is approximately $O(N_vNd)$, since kernel similarities between validation and training samples need to be computed. Therefore, the computational cost of one candidate evaluation can be approximately expressed as

$$O\left(N^2d + N^3 + N_vNd\right).$$

Given a population size P and maximum iteration number $t_{\max}$, the overall computational cost of SKELM can be written as

$$O\left(Pt_{\max}\left(N^2d + N^3 + N_vNd\right)\right).$$

This indicates that SKELM introduces additional optimization overhead compared with the original KELM, since KELM training and validation need to be repeatedly performed for multiple candidate solutions. Therefore, the proposed method is more suitable for small- and medium-scale datasets. For large-scale datasets, scalable kernel approximation, reduced candidate evaluation, early stopping, or parallel computing strategies may be required.

5.6. Discussion on parameter effects

The performance of SKELM is affected by two groups of parameters. The first group consists of the KELM model parameters, namely the regularization parameter C and the RBF kernel width σ. These two parameters directly determine the generalization ability of KELM. Specifically, C controls the trade-off between empirical fitting error and model regularization. An excessively large value of C may lead to overfitting, while an excessively small value may cause underfitting. The RBF kernel width σ determines the scale of nonlinear similarity measurement among samples. A very small σ may make the classifier overly sensitive to local variations, whereas a very large σ may weaken the discriminative ability of the kernel function.

The second group consists of the control parameters of the Snake Optimizer, including the population size P, the maximum number of iterations $t_{\max}$, and the behavioral control factors used in the exploration and exploitation phases. These parameters affect the search efficiency and convergence behavior of SO. A larger population size or more iterations may increase the probability of finding a better parameter pair (C, σ), but they also increase computational cost. Conversely, too small a population size or too few iterations may result in insufficient search and premature convergence.

Notably, this subsection provides a qualitative discussion of parameter effects rather than a full parameter sensitivity experiment. In this study, the optimization settings were fixed across all compared metaheuristic-based KELM methods to ensure a controlled comparison. Specifically, the population size was set to 30, the maximum number of iterations was set to 100, and the search

dimension was fixed at 2. Under these settings, the performance differences among LSOKELM, SSAKELM, and SKELM mainly reflect their optimization mechanisms under the adopted configuration. A systematic sensitivity analysis involving different population sizes, iteration numbers, and search ranges will be considered in future work.

5.7. Practical advantages and limitations

The proposed SKELM has several practical advantages for nonlinear classification tasks. First, it reduces the dependence on manual parameter tuning by automatically searching for the regularization parameter C and the RBF kernel width σ. This is important because inappropriate parameter settings may lead to overfitting, underfitting, or unstable classification performance. Second, SKELM may improve the stability of KELM parameter selection by using the adaptive exploration-exploitation mechanism of SO. The food quantity and environmental temperature control factors help the optimizer explore the global search space and refine promising parameter regions. Third, the compact two-dimensional encoding strategy, in which each individual represents a candidate parameter pair (C, σ), makes the optimization process simple, interpretable, and easy to implement.

From an application perspective, SKELM is suitable for small- to medium-scale nonlinear classification problems where model accuracy, training efficiency, and parameter stability are important. Potential application scenarios include medical diagnosis, fault detection, intrusion detection, financial risk assessment, quality inspection, and other classification tasks with nonlinear data distributions. Since the proposed method can automatically identify a suitable parameter pair, it is particularly useful when prior knowledge about optimal KELM parameter settings is unavailable.

However, SKELM also has several limitations. First, since KELM relies on kernel matrix computation, the computational cost may increase rapidly when the number of training samples becomes large. This limits the direct application of SKELM to large-scale datasets without kernel approximation or acceleration strategies.Second, the current fitness function is based on validation accuracy. Although accuracy is intuitive and widely used, it may be insufficient for highly imbalanced datasets, where minority-class performance is more important. In such cases, macro-F1, balanced accuracy, sensitivity, specificity, or AUC-based fitness functions may be more appropriate.

Third, highly sparse data may reduce the reliability of the RBF kernel matrix. When most feature values are zero or near zero, pairwise kernel similarities may become less informative, and the optimizer may select parameter values that fit the validation subset but generalize poorly to unseen data. Possible mitigation strategies include sparse-data preprocessing, feature selection, dimensionality reduction, alternative kernel functions, and sparsity-aware fitness functions.

Fourth, the present study compares SKELM mainly with KELM, LSOKELM, and SSAKELM. Although this comparison provides preliminary evidence for the feasibility of the SO-based parameter search strategy, stronger baselines such as grid-search KELM, random-search KELM, PSO-KELM, GA-KELM, and RBF-SVM should be included in future work. In addition, to provide a more rigorous assessment of the proposed method, future studies should report dataset-level standard deviations, confidence intervals, statistical significance tests, convergence curves, runtime comparisons, and additional evaluation metrics.

Future work will investigate scalable kernel approximation strategies to reduce computational cost on large-scale datasets. Meanwhile, imbalance-aware and sparsity-aware fitness functions will be designed to improve robustness under imbalanced or sparse data conditions. Sparse-data preprocessing, feature selection, dimensionality reduction, adaptive parameter-bound selection, and multi-objective optimization strategies will also be explored to further improve the practical applicability of SKELM in complex real-world scenarios.

5.8. Summary of experimental findings

In summary, the experimental results provide preliminary evidence that the proposed SKELM can achieve competitive classification performance on the selected UCI benchmark datasets. Compared with the original KELM, SKELM improves the reported accuracy on most datasets by automatically searching for the regularization parameter and RBF kernel width. Compared with LSOKELM and SSAKELM, SKELM alsoyields competitive performance in several datasets under the same population size, maximum iteration number, and search ranges.

However, the current experimental evidence should be interpreted with caution. The reported results are mainly based on classification accuracy. Additional metrics such as macro-F1, balanced accuracy, sensitivity, specificity, and AUC

are not included. Moreover, statistical significance tests, convergence curves, runtime comparisons, and stronger baseline methods are not provided in the present version. Therefore, the current findings support the feasibility and potential usefulness of SKELM, but they do not fully establish its general superiority over all existing parameter optimization or kernel-based classification methods.

6. Conclusion

This paper investigated the parameter sensitivity problem of Kernel Extreme Learning Machine (KELM) in nonlinear classification tasks. Since the performance of KELM is strongly affected by the regularization parameter C and the RBF kernel width σ, this study introduced a Snake Optimizer-based KELM framework, termed SKELM, to automatically search for a suitable combination of these two coupled parameters. The main purpose of this work is not to modify the original KELM learning mechanism, but to examine whether the Snake Optimizer can serve as a useful alternative search strategy for KELM parameter selection.

Experiments were conducted on ten UCI benchmark datasets under controlled optimization settings. The results indicate that SKELM obtained the unique best accuracy on five datasets, including Zoo, Heart, WDBC, P_gene, and Haberman, and ties for the best accuracy on Iris, Soybean, and Glass. In the reported convergence and stability comparison, SKELM required 35 average iterations, while LSOKELM and SSAKELM required 42 and 68 average iterations, respectively. SKELM also reported a standard deviation of 0.96%, which was lower than those of the compared methods in the current experimental records. These numerical results suggest that the SO-based search mechanism has the potential to improve KELM parameter selection under the adopted experimental setting.

The significance of this study is to provide a simple and interpretable SO-driven parameter optimization framework for KELM. By encoding each snake individual as a two-dimensional parameter pair (C, σ) and using validation accuracy as the fitness function, the proposed method reduces the need for manual parameter tuning and offers a practical optimization procedure for small- and medium-scale nonlinear classification tasks. The results also suggest that the adaptive exploration and exploitation mechanism of SO may be useful for searching the coupled parameter space of KELM.

Nevertheless, the conclusions of this study should be interpreted cautiously. The current experiments mainly rely on classification accuracy and do not include additional metrics such as macro-F1, balanced accuracy, sensitivity, specificity, or AUC. Statistical significance tests, confidence intervals, detailed convergence curves, and runtime comparisons are not provided either. In addition, the comparison is limited to KELM, LSOKELM, and SSAKELM, while stronger baselines such as grid-search KELM, random-search KELM, PSO-KELM, GA-KELM, and RBF-SVM require further investigation.

Future work will focus on strengthening the experimental validation of SKELM. More comprehensive comparisons with additional baseline methods, dataset-level standard deviations, statistical significance tests, convergence curves, runtime analysis, and multiple evaluation metrics should be included. Moreover, scalable kernel approximation, sparse-data preprocessing, feature selection, dimensionality reduction, imbalance-aware fitness functions, and sparsity-aware optimization strategies will be considered for improving the applicability of SKELM to large-scale, imbalanced, and highly sparse datasets.

Acknowledgments

The authors would like to thank Zhejiang Dongfang Polytechnic for its support.

Funding

This work was supported by the Institutional Research Project of Zhejiang Dongfang Polytechnic (Grant No. DF2025YB51) and the Scientific Research Fund of the Zhejiang Provincial Education Department, China (Grant No. Y202456182).

Conflict of interest

The authors declare they have no competing interests.

Author contributions

Conceptualization: Qi Diao and WengHowe Chan.
Data curation: Hao Long Yang and Qi Diao.
Formal analysis: Hao Long Yang and Qi Diao.
Funding acquisition: Qi Diao.
Investigation: Qi Diao.
Methodology: Qi Diao.
Software:Hao Long Yang and Qi Diao.
Supervision: Qi Diao and WengHowe Chan.
Validation: Qi Diao and Hao Long Yang.
Writing–original draft: Hao Long Yang and Qi Diao.

Writing–review and editing: WengHowe Chan and Hao Long Yang.

Availability of data

Not applicable.

AI tools statement

During the preparation of this manuscript, the authors used ChatGPT-4.0 and Qwen (Tongyi Chat) to enhance grammatical accuracy and improve the fluency of expression. All AI-assisted content was carefully reviewed and revised by the authors, who take full responsibility for the final version of the manuscript.

References

1. Mohammed A, Kora R. A comprehensive review on ensemble deep learning: opportunities and challenges. *J King Saud Univ Comput Inf Sci.* 2023;35(2):757-774. https://doi.org/10.1016/j.jksuci.2023.01.014
2. Wang H, Li W. Fast ramp fraction loss SVM classifier with low computational complexity for pattern classification. *Neural Netw.* 2025;184:107087. https://doi.org/10.1016/j.neunet.2024.107087
3. Taloba AI, Matoog RT. Detecting respiratory diseases using machine learning-based pattern recognition on spirometry data. *Alex Eng J.* 2025;113:44-59. https://doi.org/10.1016/j.aej.2024.11.009
4. Chhillar I, Singh A. An improved soft voting-based machine learning technique to detect breast cancer utilizing effective feature selection and SMOTE-ENN class balancing. *Discov Artif Intell.* 2025;5:4. https://doi.org/10.1007/s44163-025-00224-w
5. Guevara-Reyes R, Ortiz-Garcés I, Andrade R, Cox-Riquetti F, Villegas-Ch W. Machine learning models for academic performance prediction: interpretability and application in educational decision-making. *Front Educat.* 2025;10:1632315. https://doi.org/10.3389/feduc.2025.1632315
6. Cortes C, Vapnik V. Support-vector networks. *Mach Learn.* 1995;20(3):273-297. https://doi.org/10.1007/BF00994018
7. Przymus P, Rykaczewski K, Martín-Segura A, *et al.* Deep learning in microbiome analysis: a comprehensive review of neural network models. *Front Microbiol.* 2025;15:1516667. https://doi.org/10.3389/fmicb.2024.1516667
8. Kayıran HF. Machine learning-assisted thermoelastic analysis of fiber-reinforced composite rotating disks. *Intell Syst Res Appl J.* 2025;1:1-12. https://doi.org/10.59543/ctg7te95
9. Musbah J, Badi I. LLM-assisted virtual expert weight elicitation in pharmaceutical supply chains: a Z-number multi-agent framework. *Intell Syst Res Appl J.* 2026;2:27-39. https://doi.org/10.59543/mmhqdg22
10. Tang Z, Wang S, Chai X, Cao S, Ouyang T, Li Y. Auto-encoder–extreme learning machine model for boiler NO_x emission concentration prediction. *Energy.* 2022;256:124552. https://doi.org/10.1016/j.energy.2022.124552
11. Wang J, Lu S, Wang SH, *et al.* A review on extreme learning machine. *Multimed Tools Appl.* 2022;81:41611-41660. https://doi.org/10.1007/s11042-021-11007-7
12. Wang C, Peng G, De Baets B. Embedding metric learning into an extreme learning machine for scene recognition. *Expert Syst Appl.* 2022;203:117505. https://doi.org/10.1016/j.eswa.2022.117505
13. Vásquez-Coronel JA, Mora M, Vilches K. A review of multilayer extreme learning machine neural networks. *Artif Intell Rev.* 2023;56(11):13691-13742. https://doi.org/10.1007/s10462-023-10478-4
14. Diao Q, Junaidi A, Chan WH, Zain AM, Yang HL. SBOA: a novel heuristic optimization algorithm. *Baghdad Sci J.* 2024;21(2 suppl):764. https://doi.org/10.21123/bsj.2024.9766
15. Han E, Ghadimi N. Model identification of proton-exchange membrane fuel cells based on a hybrid convolutional neural network and extreme learning machine optimized by improved honey badger algorithm. *Sustain Energy Technol Assess.* 2022;52:102005. https://doi.org/10.1016/j.seta.2022.102005
16. Lv L, Wang W, Zhang Z, Liu X. A novel intrusion detection system based on an optimal hybrid kernel extreme learning machine. *Knowl Based Syst.* 2020;195:105648. https://doi.org/10.1016/j.knosys.2020.105648
17. Huang GB, Zhou H, Ding X, Zhang R. Extreme learning machine for regression and multiclass classification. *IEEE Trans Syst Man Cybern B Cybern.* 2012;42(2):513-529. https://doi.org/10.1109/TSMCB.2011.2168604
18. Liu J, Xu H, Peng X, Wang J, He C. Reliable composite fault diagnosis of hydraulic systems based on linear discriminant analysis and multi-output hybrid kernel extreme learning machine. *Reliab Eng Syst Saf.* 2023;234:109178. https://doi.org/10.1016/j.ress.2023.109178
19. Mohanty F, Rup S, Dash B, Majhi B, Swamy MNS. An improved scheme for digital mammogram classification using weighted chaotic salp swarm algorithm-based kernel extreme learning machine. *Appl Soft Comput.* 2020;91:106266. https://doi.org/10.1016/j.asoc.2020.106266
20. Zhang Y, Wang Y, Zhou G, Jin J, Wang B, Wang X, *et al.* Multi-kernel extreme learning machine for EEG classification in brain–computer interfaces. *Expert Syst Appl.* 2018;96:302-310. https://doi.org/10.1016/j.eswa.2017.12.015
21. Maharajan C, Sowmiya C, Xu C. Lagrange stability of inertial type neural networks: a

Lyapunov-Krasovskii functional approach. *Evol Syst.* 2025;16:63. https://doi.org/10.1007/s12530-025-09681-1
22. Cui Q, Xu C, Xu Y, *et al.* Bifurcation and controller design of 5D BAM neural networks with time delay. *Int J Numer Model Electron Netw Devices Fields.* 2024;37(6):e3316. https://doi.org/10.1002/jnm.3316
23. Xu C, Lin J, Zhao Y, *et al.* New results on bifurcation for fractional-order octonion-valued neural networks involving delays. *Netw Comput Neural Syst.* 2025;36(3):545-597. https://doi.org/10.1080/0954898X.2024.2332662
24. Xu C, Zhao Y, Lin J, *et al.* Bifurcation investigation and control scheme of fractional neural networks owning multiple delays. *Comput Appl Math.* 2024;43:186. https://doi.org/10.1007/s40314-024-02718-2
25. Gad AG. Particle swarm optimization algorithm and its applications: a systematic review. *Arch Comput Methods Eng.* 2022;29:2531-2561. https://doi.org/10.1007/s11831-021-09694-4
26. Tang J, Liu G, Pan Q. A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends. *IEEE/CAA J Autom Sin.* 2021;8(10):1627–1643. https://doi.org/10.1109/JAS.2021.1004129
27. Wang D, Tan D, Liu L. Particle swarm optimization algorithm: an overview. *Soft Comput.* 2018;22:387-408. https://doi.org/10.1007/s00500-016-2474-6
28. Katoch S, Chauhan SS, Kumar V. A review on genetic algorithm: past, present, and future. *Multimed Tools Appl.* 2021;80:8091-8126. https://doi.org/10.1007/s11042-020-10139-6
29. Awadallah MA, Al-Betar MA, Doush IA, *et al.* Recent versions and applications of sparrow search algorithm. *Arch Comput Methods Eng.* 2023;30:2831-2858. https://doi.org/10.1007/s11831-023-09887-z
30. Liu J, Li D, Wu Y, Liu D. Lion swarm optimization algorithm for comparative study with application to optimal dispatch of cascade hydropower stations. *Appl Soft Comput.* 2020;87:105974. https://doi.org/10.1016/j.asoc.2019.105974
31. Chen H, Miao F, Chen Y, Xiong Y, Chen T. A hyperspectral image classification method using multifeature vectors and optimized KELM. *IEEE J Sel Top Appl Earth Obs Remote Sens.* 2021;14:2781-2795. https://doi.org/10.1109/JSTARS.2021.3059451
32. Yue Y, Cao L, Chen H, Chen Y, Su Z. Towards an optimal KELM using the PSO–BOA optimization strategy with applications in data classification. *Biomimetics.* 2023;8(3):306. https://doi.org/10.3390/biomimetics8030306
33. Senapati T, Sarkar A, Chen G. Enhancing healthcare supply chain management through artificial intelligence-driven group decision-making with Sugeno-Weber triangular norms in a dual hesitant q-rung orthopair fuzzy context. *Eng Appl Artif Intell.* 2024;135:108794. https://doi.org/10.1016/j.engappai.2024.108794
34. Khan MSA, Jan SU, Jan R, Senapati T, Moslem S. Complex interval-valued intuitionistic fuzzy decision support system with application to COVID-19 healthcare facilities. *Complex Intell Syst.* 2023;9:7103-7132. https://doi.org/10.1007/s40747-023-01090-8
35. Kakati P, Quek SG, Selvachandran G, Senapati T, Chen G. Analysis and application of rectified complex t-spherical fuzzy Dombi-Choquet integral operators for diabetic retinopathy detection through fundus images. *Expert Syst Appl.* 2024;243:122724. https://doi.org/10.1016/j.eswa.2023.122724
36. Hashim F, Hussain K, Houssein EH. Snake optimizer: a novel meta-heuristic optimization algorithm. *Knowl Based Syst.* 2022;242:108320. https://doi.org/10.1016/j.knosys.2022.108320
37. Huang GB, Zhu QY, Siew CK. Extreme learning machine: Theory and applications. *Neurocomputing.* 2006;70(1–3):489–501. https://doi.org/10.1016/j.neucom.2005.12.126
38. Nair NK, S A. Approximate deep kernel learning in supervised extreme learning machines. *Pattern Anal Applic.* 2026;29:97. https://doi.org/10.1007/s10044-026-01662-7
39. Zheng K, Liu H, Li B. Improved Snake Optimization Algorithm for global optimization and engineering applications. *Sci Rep.* 2025;15:18171. https://doi.org/10.1038/s41598-025-01299-2
40. Kelly M, Longjohn R, Nottingham K.The UCI Machine Learning Repository. Accessed May 12, 2026. https://archive.ics.uci.edu/
41. Michalski RS, Chilausky RL. Soybean (Large). UCI Machine Learning Repository. Published 1988. https://doi.org/10.24432/C5JG6Z
42. Noordewier MO, Towell GG, Shavlik JW. Molecular Biology (Promoter Gene Sequences). UCI Machine Learning Repository. Published 1990. https://doi.org/10.24432/C5S01D

Haolong Yang *was born in Jinan, China on March 6, 2000. He received his MSc in Computer Science from Boston University (2024), and BEng in Software Engineering from Chongqing University (2022).His research interests include Machine Learning, Computer Vision and Artificial Intelligence.*
https://orcid.org/0009-0001-9775-8285.

Qi Diao *holds a Master' s degree and is currently pursuing his Ph.D. at Universiti Teknologi Malaysia. His research interests include Machine Learning, Computer Vision, Artificial Intelligence, and Swarm Intelligence Algorithms. He has participated in several research projects focusing on the development of intelligent systems and their real-world applications.*
https://orcid.org/0000-0002-8187-9461.

WengHowe Chan *is a senior lecturer at the Faculty of Computing and a research fellow at the UTM Big Data Centre of Universiti Teknologi Malaysia, where he also received his Ph.D. in Computer Science in 2016. His research expertise lies in the areas of machine learning algorithms for computational biology and cancer bioinformatics, computational intelligence for advanced analytics, and advanced data storage design for data analytics. He has also applied his skills to various case studies such as sentiment analysis, talent analytics, learning analytics, and analysis of different biological data.*


https://orcid.org/0000-0003-0612-3661.

Appendix A. List of abbreviations

This appendix lists the main abbreviations used in this paper to improve readability and ensure consistent terminology throughout the manuscript.

List of abbreviations used in this paper

Abbreviation	Full term
ELM	Extreme Learning Machine
KELM	Kernel Extreme Learning Machine
SKELM	Snake Optimizer-based Kernel Extreme Learning Machine
SO	Snake Optimizer
RBF	Radial Basis Function
PSO	Particle Swarm Optimization
GA	Genetic Algorithm
SSA	Sparrow Search Algorithm
LSO	Lion Swarm Optimization
LSOKELM	Lion Swarm Optimization–based Kernel Extreme Learning Machine
SSAKELM	Sparrow Search Algorithm–based Kernel Extreme Learning Machine
UCI	University of California Irvine Machine Learning Repository

An International Journal of Optimization and Control: Theories & Applications (https://accscience.com/journal/ijocta)