High-Precision Light Trucks Fuel Consumption Prediction using XGBoost- Improved Arithmetic Optimization Algorithm-DeepESN

Yishuai Li[1], Wei Xu[2], Chengliang Liao[2], Kuan Kei HOI[3*], Liaodong Nie[3],
Ning Wang[3], Jingming Wu[2], Yiming Hu[1], Dequan Zeng[1]

[1]Nanchang Automotive Institute of Intelligence & New Energy, Nanchang 330000, China
[2]Jiangxi Isuzu Motors Co., Ltd., Nanchang 330000, China
[3]School of Automotive Studies, Tongji University, Shanghai 201800, China

liyishuai@naiine.com(Yishuai Li); 383089729@qq.com(Wei Xu); liao.chengliang@jiangxi-isuzu.cn(Chengliang Liao); *2230015@tongji.edu.cn(Kuan Kei HOI); 2311436@tongji.edu.cn(Liaodong Nie); wangning@tongji.edu.cn(Ning Wang); wu.jingming@jiangxi-isuzu.cn(Jingming Wu); huyiming@naiine.com(Yiming Hu); zdq1610849@126.com(Dequan Zeng)

Abstract. With the increasing demand for light-duty diesel trucks in urban and surrounding areas due to the growth in China's road freight and urban distribution, the issues of fuel consumption and environmental emissions have become more severe. This paper proposes a fuel consumption prediction optimization model that combines XGBoost for selecting key fuel consumption features and the DeepESN algorithm for prediction. By incorporating Tent Chaotic Mapping, Nonlinear Weight Factor, Cauchy Mutation Strategy, and Sparrow Alarming Strategy, the Improved Arithmetic Optimization Algorithm (IPOA) is employed to optimize the hyperparameters of DeepESN. The model is validated using T-BOX data from 150 light trucks, and the results indicate that the XGBoost-IPOA-DeepESN model outperforms other comparative models in terms of prediction accuracy, providing a reference for implementing efficient energy use strategies.

Keywords: Vehicle Fuel Consumption Prediction; Light Trucks; DeepESN; IPOA

1 Introduction

With the rapid development of the social economy, the volume of road freight transportation in China has been increasing annually. By 2022, the road freight volume reached 37.12 billion tons, and the transportation turnover soared to 22,612.18 billion ton-kilometers [1]. This trend signifies greater energy consumption and environmental emissions, posing challenges to sustainable development. The rapid growth of e-commerce has further increased urban freight distribution. From 2013 to 2022, the volume

G. Zhao et al. (eds.), *Proceedings of the 2024 7th International Symposium on Traffic Transportation and Civil Architecture (ISTTCA 2024)*, Advances in Engineering Research 241,
https://doi.org/10.2991/978-94-6463-514-0_83

of postal delivery services in China grew from 9.18 billion to 139.1 billion pieces [2,3]. This has led to a rising demand for freight vehicles, especially light-duty diesel trucks used for urban transportation, becoming a major component of road traffic in many cities.

Sustainable development is now a prerequisite for industrial operations worldwide, especially in transportation. Most vehicles still rely on non-renewable energy sources like petroleum, producing greenhouse gases and other harmful emissions. With the depletion of petroleum resources and deteriorating environmental quality, reducing vehicle fuel consumption is urgent. Detailed analysis and accurate prediction of fuel consumption for freight vehicles are crucial. This not only helps implement more efficient energy use strategies but also contributes to reducing overall fuel consumption, lowering environmental emissions, and promoting sustainable development in the transportation industry.

With the advancement of data mining and information technology, machine learning methods have been widely applied in fuel consumption prediction [4-5]. Compared to traditional physical models, machine learning models efficiently handle large-scale data, especially with real-time data from onboard T-Box devices, allowing for rapid updates and iterations. Mainstream models include support vector machine (SVM) [6], Random Forest (RF)[7], Gradient Boosting Decision Tree (GBDT)[8], and neural networks like Backpropagation (BP)[9] and Long Short-Term Memory (LSTM)[10], etc. For example, Zhao et al. used BP neural networks and Principal Component Analysis to predict fuel consumption on urban expressway [11]. Wickramanayake et al. used RF, GB, and neural networks with GPS and fuel sensor data for long-distance bus fuel prediction [12]. Yao et al. combined smartphone-collected driving data and OBD fuel data to predict taxi fuel consumption using BP, Support Vector Regression, and RF [13]. Kanarachos et al. utilized RNNs to predict instant fuel consumption under various driving conditions, but faced issues with gradient problems [14]. Wang et al. used an LSTM neural network to accurately predict vehicle fuel consumption, though empirical hyperparameter selection affected accuracy [15].

Considering that prediction accuracy is influenced by hyperparameters [16], heuristic optimization algorithms have been proposed to improve model accuracy. Niu et al. enhanced SVR prediction accuracy using Artificial Fish School Algorithm(AFSA) [17]. Gu et al. used PSOGA to optimize SVM hyperparameters for mining truck fuel consumption, improving performance but struggling with local optima in high-dimensional problems [18]. Han et al. used Improved Grey Wolf Optimization (IGWO) to optimize LSTM hyperparameters for ship fuel consumption, with XGBoost for feature selection [19].

The Echo State Network (ESN) is a new type of neural network with strong data processing capabilities, though its application in fuel consumption prediction is still limited. Given the complex nature of vehicle fuel consumption, this study first utilizes XGBoost to select key features affecting fuel consumption. A data-driven model is then constructed based on the DeepESN algorithm to evaluate its prediction performance. To further enhance model performance, an Improved Arithmetic Optimization Algorithm (IPOA) is used to optimize DeepESN hyperparameters. Practical case validation results show that the proposed XGBoost-IPOA-DeepESN optimization model achieves the highest prediction accuracy compared to other models.

2 Fundamental Principles

2.1 Deep Echo State Network

The DeepESN is a type of deep recurrent neural network model designed for processing time series data. It extends the traditional ESN by introducing a hierarchical organization of recurrent layers, enabling more efficient processing of temporal information and multi-time scale representation [20-21].

The basic structure of DeepESN includes five main concepts: state, input, output, reservoir layers, and leakage rate [22]. The state (x) represents the internal state of the network at a given time point, the input (u) represents the external input signal received by the network, and the output (y) is computed through the network's state and the output layer weight matrix. Reservoir layers are the core components of DeepESN, consisting of multiple hierarchical recurrent units where the output of each layer serves as the input to the next. The leakage rate (a) controls the mix of new and old information during state updates.

At each time step, the state update equations for DeepESN are as follows:

For the first layer, the state update formula is:

$$x^{(1)}(t) = (1 - a^{(1)})x^{(1)}(t-1) + a^{(1)}f(W^{(1)}u(t) + \widehat{W}^{(1)}x^{(1)}(t-1)) \tag{1}$$

For the i -th layer (i>1), the state update formula is:

$$x^{(i)}(t) = (1 - a^{(i)})x^{(i)}(t-1) + a^{(i)}f(W^{(i)}x^{(i-1)}(t) + \widehat{W}^{(i)}x^{(i)}(t-1)) \tag{2}$$

where $W^{(1)}$ is the input weight matrix, $W^{(i)}$ is the inter-layer connection weight matrix, $\widehat{W}^{(i)}$ is the recurrent weight matrix, $a^{(i)}$ is the leakage rate, and f is the activation function.

DeepESN exhibits several critical characteristics. First, the recurrent weights remain fixed after initialization, ensuring network stability through the Echo State Property (ESP). Second, the hierarchical reservoir layer structure enables the network to develop multi-time scale representations within its internal states. Additionally, DeepESN can be simplified into a layered version of a single-layer ESN by applying constraints, such as removing certain inter-layer connections, which affect how different sub-parts process temporal information [23].

For output computation, the states of all layers at each time step are used as inputs to the output layer, calculated as follows:

$$y(t) = W_{out}x(t) \tag{3}$$

where W_{out} is the readout weight matrix.

The innovation of DeepESN lies in its hierarchical recurrent structure and automatic layer selection method. Compared to traditional ESN, DeepESN captures and represents multi-time scale information more efficiently through its layered recurrent structure.

2.2 Pelican Optimization Algorithm

The Pelican Optimization Algorithm (POA) is a novel nature-inspired stochastic optimization algorithm [24]. The design of POA is inspired by the natural behavior of pelicans during hunting. Pelicans dive from a height to catch prey after identifying its location and then spread their wings on the water surface to push fish towards shallow water for easier capture. This algorithm mimics this process by updating the positions of population members in the search space to solve optimization problems.

The basic structure of POA includes Search Agents, Search Space, Objective Function, and Initial Position. Search Agents represent candidate solutions in the search space [25]. The search space is the solution space where candidate solutions are searched and optimized. The objective function evaluates the quality of candidate solutions. The initial positions of the population members are randomly initialized within the problem's bounds.

POA achieves candidate solution updates and optimization through two main phases: approaching the prey (exploration phase) and spreading wings on the water surface (exploitation phase). In the exploration phase, population members simulate pelicans identifying prey location and moving towards it, enhancing the algorithm's exploration capability. The specific state update formula is:

$$x_{i,j}^{P1} = \begin{cases} x_{i,j} + \text{rand} \cdot (p_j - I \cdot x_{i,j}), & F_p < F_i \\ x_{i,j} + \text{rand} \cdot (x_{i,j} - p_j), & \text{else} \end{cases} \tag{4}$$

where $x_{i,j}^{P1}$ is the updated value of the i-th candidate solution in the j-th dimension, p_j is the prey's position in the j-th dimension, I is a random number, F_p and F_i are the objective function values of the prey and the candidate solution, respectively.

In the exploitation phase, population members simulate pelicans spreading their wings on the water surface to push fish towards shallow water, enhancing the algorithm's local search capability. The specific state update formula is:

$$x_{i,j}^{P2} = x_{i,j} + R \cdot \left(1 - \frac{t}{T}\right) \cdot (2 \cdot \text{rand} - 1) \cdot x_{i,j} \tag{5}$$

where $x_{i,j}^{P2}$ is the updated value of the i-th candidate solution in the j-th dimension, R is a constant, t is the iteration counter, and T is the maximum number of iterations. The coefficient $R \cdot \left(1 - \frac{t}{T}\right)$ represents the neighborhood radius of the population members, which decreases as the number of iterations increases.

POA performs effective updates in each phase. If the objective function value of the new position is better than the current value, the new position is accepted. The specific formula is:

$$x_i = \begin{cases} x_i^{P1}, & F_i^{P1} < F_i \\ x_i^{P2}, & F_i^{P2} < F_i \\ x_i, & \text{else} \end{cases} \tag{6}$$

where x_i^{P1} and x_i^{P2} are the new positions after the exploration and exploitation phases, respectively, and F_i^{P1} and F_i^{P2} are their respective objective function values.

3 Optimization Algorithm Improvement.

3.1 Algorithm Improvement.

POA simulates the hunting behavior of pelicans, has been proven to be highly efficient in solving various optimization problems [26]. However, POA may encounter issues such as local optima and slow convergence when dealing with complex and high-dimensional problems. To address these challenges, this paper proposes an Improved Pelican Optimization Algorithm (IPOA) to enhance the algorithm's global search capability and convergence speed. The IPOA process is illustrated in Fig. 1.

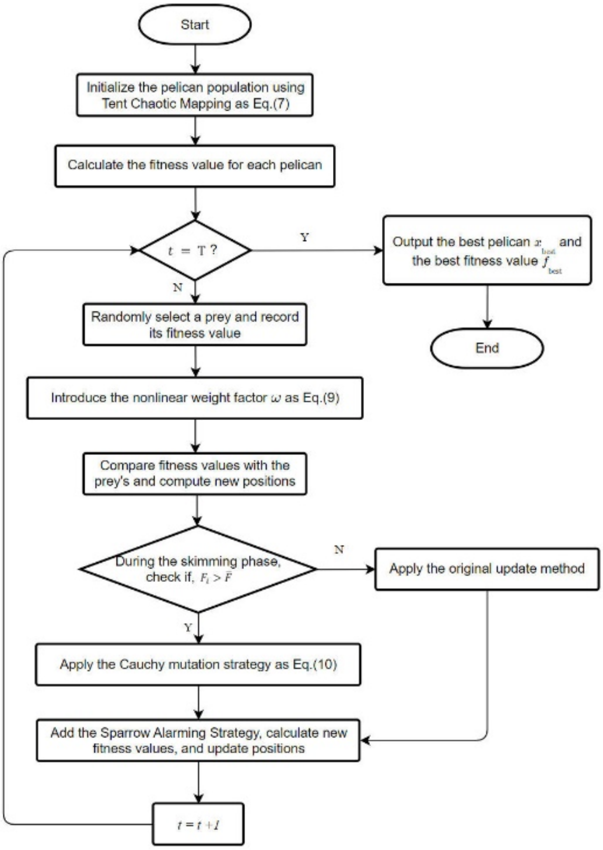
Start
Initialize the pelican population using Tent Chaotic Mapping as Eq.(7)
Calculate the fitness value for each pelican
t = T ?
Y
Output the best pelican x_best and the best fitness value f_best
End
N
Randomly select a prey and record its fitness value
Introduce the nonlinear weight factor ω as Eq.(9)
Compare fitness values with the prey's and compute new positions
During the skimming phase, check if, F_i > F
N
Apply the original update method
Y
Apply the Cauchy mutation strategy as Eq.(10)
Add the Sparrow Alarming Strategy, calculate new fitness values, and update positions
t = t + 1

Fig. 1. IPOA flow chart

Tent Chaotic Mapping.

Since the initial population in basic POA is randomly generated, it cannot ensure a uniform distribution of individuals in the search space, which affects the search speed and optimization performance [27]. In the initialization process of IPOA, Tent chaos mapping is introduced to increase the ergodicity of the initial population. The specific formula is:

$$x_{i,j} = l_j + (u_j - l_j) \cdot z_i(\alpha), i = 1,2,\dots,N, j = 1,2,\dots,m, \tag{7}$$

The Tent mapping is defined as follows:

$$z_{i+1} = \begin{cases} \frac{z_i}{\alpha}, & | \quad z_i \in [0,\alpha) \\ \frac{1-z_i}{1-\alpha}, & | \quad z_i \in [\alpha,1) \end{cases} \tag{8}$$

where $x_{i,j}$ is the position of the i-th pelican in the j-th dimension, l_j and u_j are the lower and upper bounds of the variable, $z_i(\alpha)$ is the chaotic sequence, and α is a constant.

Nonlinear Weight Factor.

To balance the local exploitation and global exploration abilities of the algorithm, IPOA introduces a nonlinear weight factor ω during the phase of moving towards the prey. This factor adjusts the relevance of the pelican's position update with the current position information. The formula is as follows:

$$x_{i,j}^{P1} = x_{i,j} + \omega(t) \cdot \text{rand} \cdot (p_j - I \cdot x_{i,j}) \tag{9}$$

where ω is smaller at the early iterations to favor global search, and increases over time to enhance local search capabilities.

Cauchy Mutation Strategy.

During the surface skimming phase, the Cauchy mutation strategy is introduced. Each iteration compares the current pelican's fitness value with the population's average fitness. If the fitness value is higher than the average, the Cauchy mutation strategy is applied to increase diversity; otherwise, the original position update method is used. The formula is as follows:

$$x_{i,j}^{P2} = \begin{cases} x_{i,j} + \text{Cauchy}(\sigma), & \text{if } F_i > \bar{F} \\ x_{i,j} + \left(1 - \frac{t}{T}\right) \cdot (2 \cdot \text{rand} - 1) \cdot x_{i,j}, & \text{else} \end{cases} \tag{10}$$

where $\text{Cauchy}(\sigma)$ is the Cauchy distribution, and σ is the mutation intensity.

Sparrow Alarming Strategy.

During the surface skimming phase, a sparrow alarming strategy is added. When pelicans perceive danger, the pelicans on the edge of the group quickly move to a safe area, while those in the middle move randomly to approach other pelicans. This strategy accelerates convergence and enhances global search capabilities.

For pelicans in a dangerous state (edge individuals), the update formula is:

$$x_{i,j} = x_{i,j} + \text{rand} \cdot (x_{\text{best},j} - x_{i,j}) \tag{11}$$

where $x_{\mathrm{best},j}$ is the current optimal position in the population.

For pelicans in a safe state (middle individuals), the update formula is:

$$x_{i,j} = x_{i,j} + \mathrm{randn} \cdot (\bar{x}_j - x_{i,j}) \tag{12}$$

where *randn* is a random number from the standard normal distribution, and $\bar{x}_j$ is the average position of the population.

3.2 Performance Evaluation

To evaluate the enhancement effect of the IPOA algorithm, it was benchmarked on 13 commonly used unimodal and multimodal benchmark functions [28-30]. The standard benchmark functions are shown in Table 1, where Dim represents the dimension of the function, Range is the boundary of the function's search space, and f_{min} is the optimal value. The total number of iterations for the algorithm was set to 500, with each algorithm running 30 times on each benchmark function. The average and standard deviation of the 30 runs were calculated, and the statistical results are presented in Table 2. To validate the results, the IPOA algorithm was compared with POA [24], GWO [30], PSO [31], and GA [32].

Table 1. Benchmark functions

Function	Dim	Range	f_{min}
$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	[-100,100]	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[-10,10]	0
$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$	30	[-100,100]	0
$f_4(x) = \max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$	30	[-100,100]	0
$f_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	[-30,30]	0
$f_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30	[-100,100]	0

$f_7(x) = \sum_{i=1}^{n} i\, x_i^4 + random[0,1)$	30	[-1.28,1.28]	0
$f_8(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30	[-500,500]	-9850
$f_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	[-5.12,5.12]	0
$f_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	30	[-32,32]	0
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[-600,600]	0
$f_{12}(x) = \frac{\pi}{n}\left\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10,100,4)$ $y_i = 1 + \frac{x_i + 1}{4} \quad , u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	[-50,50]	0
$f_{13}(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\right\} + \sum_{i=1}^{n} u(x_i, 5,100,4)$	30	[-50,50]	0

Table 2. Results of benchmark functions

	IPOA		POA		GWO		PSO		GA	
	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.	Avg.	Std.
f_1	0.00E+00	0.00E+00	1.26E-210	0.00E+00	7.33E-28	7.72E-28	2.72E+02	8.88E+01	1.15E+03	1.23E+02
f_2	7.47E-170	0.00E+00	3.18E-104	9.38E-104	1.03E-16	7.42E-17	1.34E+01	4.37E+00	7.64E+00	1.36E+00
f_3	1.55E-284	0.00E+00	4.77E-212	0.00E+00	4.92E-05	1.41E-04	4.15E+03	3.13E+03	2.34E+03	6.59E+02
f_4	5.45E-164	0.00E+00	1.49E-105	2.83E-105	7.04E-07	6.99E-07	1.72E+01	4.08E+00	2.04E+01	2.20E+00
f_5	2.80E+01	6.81E-01	2.78E+01	3.88E-01	2.74E+01	1.05E+00	1.57E+04	5.89E+03	2.36E+05	1.34E+05
f_6	3.36E+00	5.56E-01	3.53E+00	2.03E-01	7.84E-01	2.75E-01	2.70E+02	5.61E+01	9.93E+02	1.69E+02

f_7	1.30E-01	2.78E-17	1.30E-01	2.78E-17	1.30E-01	1.96E-17	1.35E-01	3.20E-03	3.31E-01	1.51E-01
f_8	-6.55E+03	9.09E+02	-4.85E+03	4.54E+02	-3.80E+03	4.06E+02	-5.41E+03	9.00E+02	-6.91E+03	5.66E+02
f_9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.75E+00	5.55E+00	1.21E+02	2.52E+01	5.58E+01	1.35E+01
f_{10}	4.44E-16	0.00E+00	4.44E-16	0.00E+00	2.08E+01	7.14E-02	2.00E+01	0.00E+00	1.06E+01	5.19E+00
f_{11}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.51E-02	1.72E-02	3.28E+00	3.63E-01	8.98E+00	1.81E+00
f_{12}	-7.03E-01	1.00E-01	-6.83E-01	4.51E-02	-9.78E-01	2.06E-02	2.28E+01	2.91E+01	8.12E+01	1.50E+02
f_{13}	2.18E+00	4.09E-01	2.37E+00	2.25E-01	4.85E-01	2.10E-01	1.29E+01	4.24E+00	5.22E+01	1.34E+01

The tests on unimodal and multimodal functions are shown in Table 2, demonstrating that IPOA provides highly competitive results. IPOA effectively avoids local optima and approaches the optimal solution with fewer iterations. The algorithm outperforms all other algorithms in f_1 to f_4 and f_9 to f_{11}, with average test results of 0 and a standard deviation of 0 for f_1, f_9 and f_{11}, indicating stable convergence to the global optimum in every test. Through experiments on multiple benchmark functions and engineering design problems, IPOA has shown stronger global search capabilities and faster convergence speed compared to POA. Overall, although IPOA may not achieve the best performance on every individual test function, its overall performance still surpasses that of other tested algorithms.

4 Fuel Consumption Prediction Model Design

4.1 Real-Time Operational Data Collection

In this study, data samples were obtained from the onboard data collection terminal (T-BOX) firmware, which collects operating data from light-duty diesel trucks. Data was collected from 150 light trucks with identical technical parameters and models across various provinces and cities in China. The data includes driver behavior and vehicle condition data, sampled every 10 seconds. The data format and structure are shown in Table 3 below.

Table 3. T-BOX Data Format

Feature Variable	Unit	Feature Variable	Unit	Feature Variable	Unit	Feature Variable	Unit
Vehicle ID	-	Engine Status	-	DPF Pressure	kPa	Reagent Level	%
Collection Time	-	Engine Net Output Torque	%	SCR Upstream NOx-Sensor Output	ppm	Intake Air	kg/h
Longitude	-	Friction Torque	%	SCR Downstream NOxSensor Output	ppm	Fuel Level	%
Latitude	-	Engine RPM	rpm	SCR Inlet Temperature	°C	Endurance Mileage	km
MILStatus	-	Fuel Flow Rate	L/h	SCR Outlet Temperature	°C	Accumulative Mileage	km
Vehicle Speed	km/h	Engine Coolant Temperature	°C	Atmospheric Pressure	kPa	Urea Level	%

Given potential anomalies or missing data during collection and transmission, this study conducted data cleaning to ensure accuracy and completeness. The process involved: examining time series continuity to correct temporal inconsistencies; using appropriate methods to fill missing data; and removing data inconsistent with actual driving conditions. These steps improved the dataset quality for modeling and analysis.

4.2 Development of the XGBoost-IPOA-DeepESN Prediction Model

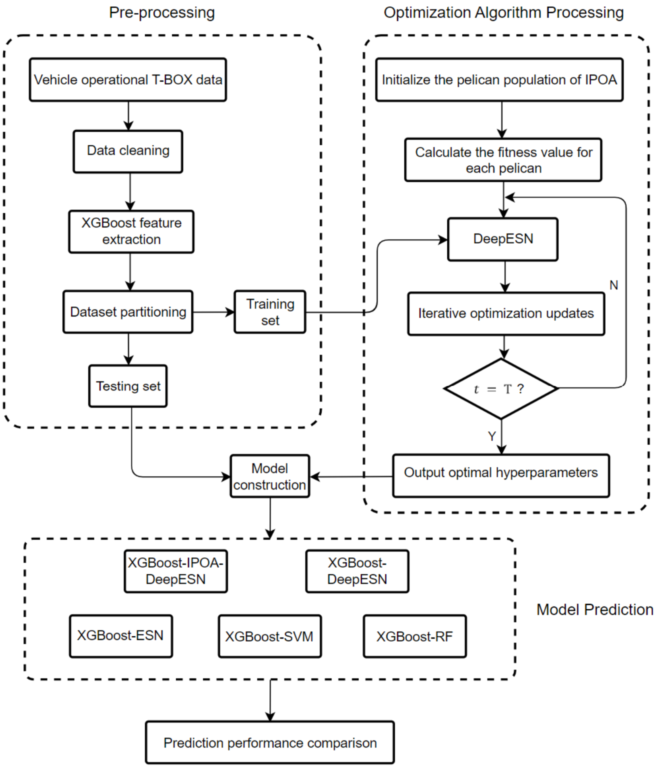
Pre-processing
Optimization Algorithm Processing
Vehicle operational T-BOX data
Data cleaning
XGBoost feature extraction
Dataset partitioning
Training set
Testing set
Initialize the pelican population of IPOA
Calculate the fitness value for each pelican
DeepESN
Iterative optimization updates
N
t = T ?
Y
Output optimal hyperparameters
Model construction
XGBoost-IPOA-DeepESN
XGBoost-DeepESN
XGBoost-ESN
XGBoost-SVM
XGBoost-RF
Model Prediction
Prediction performance comparison

Fig. 2. Flow Chart of Fuel Consumption Prediction Based on XGBoost-IPOA-DeepESN

The flow chart of fuel consumption prediction shows in Fig.2. The specific process of the chart is as follows:

Step 1: Preprocess the vehicle fuel consumption data, select key features using XGBoost, and split the data into training and testing sets with a ratio of 7:3.

Step 2: Develop the IPOA optimization algorithm incorporating Tent Chaotic Mapping, Nonlinear Weight Factor, Cauchy Mutation Strategy, and Sparrow Alarming Strategy, and set the maximum number of iterations.

Step 3: Initialize the IPOA population, calculate individual fitness values, optimize and update the hyperparameters of DeepESN, and record the position and fitness value of each pelican.

Step 4: Determine the stopping condition of the algorithm: if the current iteration count $t = T$, the algorithm stops, and the optimal DeepESN hyperparameters are output for model training; otherwise, return to Step 3 for iterative optimization.

Step 5: Use the output optimal hyperparameters to build the fuel consumption prediction model and compare the prediction results with those from DeepESN, ESN, SVM, and RF models.

4.3 Model Evaluation

This study evaluates the model's prediction performance using three key metrics: Mean Square Error (MSE), Mean Absolute Percentage Error (MAPE), and the coefficient of determination (R^2).

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{13}$$

$$\mathrm{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i-\hat{y}_i}{y_i}\right| \tag{14}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{\sum_{i=1}^{n}(y_i-\bar{y})^2} \tag{15}$$

where y_i is the actual fuel consumption for the i-th sample, $\hat{y}_i$ is the predicted fuel consumption for the i-th sample, $\bar{y}$ is the mean of the actual fuel consumption values, and n is the number of samples.

4.4 Results Validation and Analysis

The prediction results of the models are detailed in Table 4, which summarizes the accuracy performance of each fuel consumption prediction model. Comparisons were made among DeepESN, ESN, SVM, and RF prediction models to comprehensively verify the optimization effect of IPOA on DeepESN. The R^2, MSE, and MAPE of the optimized DeepESN were 0.9291, 1.5287, and 8.95%, respectively. Compared to the DeepESN model without IPOA hyperparameter optimization, the MAPE improved by 3.01%. Data from 150 experimental results were used for comparative analysis of the model's predictive performance, as shown in Fig. 3. In conclusion, the proposed integrated model is effective and enhances the accuracy of vehicle fuel consumption prediction.

Table 4. Comparison of experimental results and data

	XGBoost-IPOA-DeepESN	XGBoost-Deep-ESN	XGBoost-ESN	XGBoost-SVM	XGBoost-RF
R^2	0.9291	0.8638	0.8548	0.8252	0.8851
MSE	1.5287	1.8284	1.8729	1.9751	1.7649
MAPE	8.95%	11.96%	12.71%	14.42%	10.83%

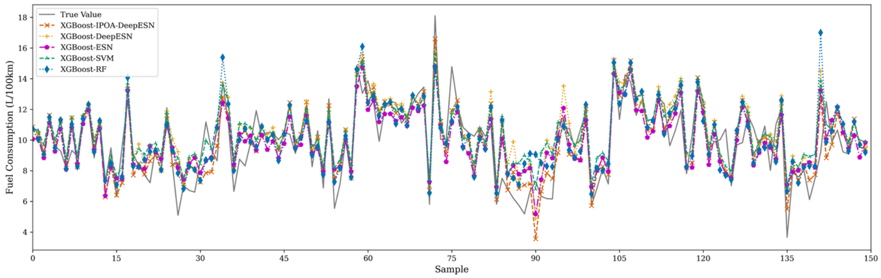

Fig. 3. Comparison of Prediction Results of Different Models

5 Conclusion

This paper integrates machine learning, meta-heuristic algorithms, and deep learning to propose a hybrid model for accurate vehicle fuel consumption prediction. The IPOA was enhanced with Tent Chaotic Mapping, Nonlinear Weight Factor, Cauchy Mutation Strategy, and Sparrow Alarming Strategy. The performance of these improvements was tested on 13 benchmark functions. Comparative results with other optimization algorithms indicate that the enhancements significantly improved the performance of the POA algorithm, providing stronger global search capabilities and higher computational accuracy. This optimization effectively improves the hyperparameter tuning of the DeepESN algorithm, thereby enhancing the model's prediction accuracy. Additionally, a comparative experiment was conducted using real-time T-BOX fuel consumption data from vehicles. The results show that the model achieved an optimal MAPE value of 8.95% and a maximum R^2 of 0.9291, demonstrating the superior performance and competitiveness of the proposed XGBoost-IPOA-DeepESN model compared to other predictive models.

The findings of this study can provide a reference for implementing efficient energy use strategies. Future work can explore different types and scales of vehicle data, considering multi-dimensional factors such as vehicle load, tire pressure, and weather conditions. Further research should seek more effective model design methods and validate and optimize the models in a wider range of application scenarios to improve the accuracy and practicality of fuel consumption predictions.

Acknowledgments

This research was supported by funding from the Nanchang Automotive Institute of Intelligence & New Energy project (Project Contract Number: TPD-TC202303-11).

Reference

1. National Bureau of Statistics of China. Statistical Communique on the National Economic and Social Development of the People's Republic of China for the Year 2022 [R/OL].

[2023−02−28]. http://www.stats.gov.cn/sj/zxfb/202302/t20230228_1919011.html.
2. Ministry of Transport of the People's Republic of China. State Post Bureau Announces 2013 Postal Industry Operation Situation [EB/OL]. [2023-01-18]. https://www.mot.gov.cn/tongjishuju/youzheng/201510/t20151014_1900997.html.
3. Ministry of Transport of the People's Republic of China. State Post Bureau Announces 2022 Postal Industry Operation Situation [EB/OL]. [2023-01-18]. https://www.mot.gov.cn/tongjishuju/youzheng/202301/t20230130_3747917.html.
4. Du, Y., Wu, J., Yang, S., & Zhou, L. (2017) Predicting vehicle fuel consumption patterns using floating vehicle data[J]. Journal of Environmental Sciences, 59, 24-29.
5. Wysocki, O., Deka, L., & Elizondo, D. (2019) Heavy duty vehicle fuel consumption modeling using artificial neural networks[J]. In 2019 25th International Conference on Automation and Computing (ICAC), 2019, September, pp. 1-6. IEEE.
6. Perrotta, F., Parry, T., & Neves, L. C. (2017) Application of machine learning for fuel consumption modelling of trucks. In 2017 IEEE International Conference on Big Data. 2017, December, pp. 3810-3815. IEEE.
7. Gong, J.; Shang, J.; Li, L.; Zhang, C.; He, J.; Ma, J. A. (2021) Comparative Study on Fuel Consumption Prediction Methods of Heavy-Duty Diesel Trucks Considering 21 Influencing Factors. Energies 2021, 14, 8106.
8. Chen Lu, Wu Hua. (2022) Design of ship fuel consumption prediction model based on GBDT. Electronic Design Engineering, 30(2): 91-95.
9. Du, Y.; Wu, J.; Yang, S.; Zhou, L. (2017) Predicting vehicle fuel consumption patterns using floating vehicle data. J. Environ. Sci., 59, 24–29.
10. Kan, Y.; Liu, H.; Lu, X.; Chen, Q. (2020) A Deep Learning Engine Power Model for Estimating the Fuel Consumption of Heavy-Duty Trucks. In Proceedings of the 2020 6th IEEE International Energy Conference (ENERGYCon), Gammarth, Tunisia, 28 September–1 October 2020; pp. 182–187.
11. Zhao X. H., Yao Y., Wu Y. P., Chen C., Rong J. (2016) Research on driving energy consumption combination prediction model based on principal component analysis and BP neural network. Journal of Transportation Systems Engineering and Information Technology, 16(5): 185-191+204.
12. Wickramanayake, S., & Bandara, H. D. (2016) Fuel consumption prediction of fleet vehicles using machine learning: A comparative study. In 2016 Moratuwa Engineering Research Conference (MERCon), 2016, April, pp. 90-95. IEEE.
13. Yao, Y., Zhao, X., Liu, C., Rong, J., Zhang, Y., Dong, Z., & Su, Y. (2020) Vehicle fuel consumption prediction method based on driving behavior data collected from smartphones[J]. Journal of Advanced Transportation, 1-11.
14. Kanarachos S, Mathew J, Fitzpatrick M E. (2019) Instantaneous vehicle fuel consumption estimation using smartphones and recurrent neural networks[J]. Expert Systems with Applications, 120: 436-447.
15. Wang Y. T., Xing B. B., Li B., L. G., Zhang X. Y. (2023) Fuel consumption prediction method for heavy-duty trucks with different acceleration driving behaviors based on Shared-LSTM. Computer and Modernization, (3): 121-126.
16. Wu J., Chen S. P., Chen X. Y., et al. (2020) Reinforcement learning for model selection and hyperparameter optimization[J]. Journal of University of Electronic Science and Technology of China, 49(2):255-261.
17. Niu X. X., Liu W. B., Nie Z. B., et al. (2019) Diesel Engine Performance Prediction Model Based on AFSA optimized SVR[J]. Ship Engineering, 41(07),44-48+79.
18. Gu Q. H., Wang Q., Jiang S., Ma P. P. (2021) Prediction of fuel consumption of open-pit mine trucks based on PSOGA-SVM. Mining Research and Development, 41(8): 161-166.

19. Han, P., Liu, Z., Sun, Z., & Yan, C. (2024) A novel prediction model for ship fuel consumption considering shipping data privacy: An XGBoost-IGWO-LSTM-based personalized federated learning approach. Ocean Engineering, 302, 117668.
20. Kim, T., & King, B. R. (2020) Time series prediction using deep echo state networks. Neural Computing and Applications, 32(23), 17769-17787.
21. Hu, H., Wang, L., & Lv, S. X. (2020) Forecasting energy consumption and wind power generation using deep echo state network. Renewable Energy, 154, 598-613.
22. Gallicchio, C., Micheli, A., & Pedrelli, L. (2018) Design of deep echo state networks. Neural Networks, 108, 33-47.
23. Sun, C., Song, M., Cai, D., Zhang, B., Hong, S., & Li, H. (2022) A systematic review of echo state networks from design to application. IEEE Transactions on Artificial Intelligence, 5(1), 23-37.
24. Trojovský, P., & Dehghani, M. (2022) Pelican optimization algorithm: A novel nature-inspired algorithm for engineering applications. Sensors, 22(3), 855.
25. Kusuma, P. D., & Prasasti, A. L. (2022) Guided pelican algorithm. International Journal of Intelligent Engineering and Systems, 15(6), 179-190.
26. SeyedGarmroudi, S., Kayakutlu, G., Kayalica, M. O., & Çolak, Ü. (2024) Improved Pelican optimization algorithm for solving load dispatch problems. Energy, 289, 129811.
27. Li, Y., Liu, Y., Lin, C., Wen, J., Yan, P., & Wang, Y. (2023) An Improved Pelican Optimization Algorithm Based on Chaos Mapping Factor. Engineering Letters, 31(4).
28. Liang J, Suganthan P, Deb K. (2005) Novel composition test functions for numerical global optimization. In: Swarm intelligence symposium, 2005. SIS 2005. Proceedings 2005 IEEE; p. 68–75.
29. Mirjalili S, Lewis A. S-shaped versus V-shaped transfer functions for binary Particle Swarm Optimization. Swarm Evolut Comput, 2013; 9:1–14.
30. Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014) Grey wolf optimizer. Advances in engineering software, 69, 46-61.
31. Marini, F., & Walczak, B. (2015) Particle swarm optimization (PSO). A tutorial. Chemometrics and Intelligent Laboratory Systems, 149, 153-165.
32. Houck, C. R., Joines, J., & Kay, M. G. (1995) A genetic algorithm for function optimization: a Matlab implementation. Ncsu-ie tr, 95(09), 1-10.