symmetry

MDPI

Article

The Orb-Weaving Spider Algorithm for Training of Recurrent Neural Networks

Anton S. Mikhalev [1,2,*], Vadim S. Tynchenko [2,3,4], Vladimir A. Nelyub [2], Nina M. Lugovaya [1], Vladimir A. Baranov [1], Vladislav V. Kukartsev [2,5,6], Roman B. Sergienko [7] and Sergei O. Kurashkin [2,4,8]

1 Scientific and Educational Software Laboratory of Informatics Department, Institute of Space and Information Technologies, Siberian Federal University, 660041 Krasnoyarsk, Russia
2 Digital Material Science: New Materials and Technologies, Bauman Moscow State Technical University, 105005 Moscow, Russia
3 Department of Technological Machines and Equipment of Oil and Gas Complex, School of Petroleum and Natural Gas Engineering, Siberian Federal University, 660041 Krasnoyarsk, Russia
4 Information-Control Systems Department, Institute of Computer Science and Telecommunications, Reshetnev Siberian State University of Science and Technology, 660037 Krasnoyarsk, Russia
5 Department of Informatics, Institute of Space and Information Technologies, Siberian Federal University, 660041 Krasnoyarsk, Russia
6 Department of Information Economic Systems, Institute of Engineering and Economics, Reshetnev Siberian State University of Science and Technology, 660037 Krasnoyarsk, Russia
7 Machine Learning Department, Gini Gmbh, 80339 Munich, Germany
8 Laboratory of Biofuel Compositions, Siberian Federal University, 660041 Krasnoyarsk, Russia
* Correspondence: asmikhalev@yandex.ru

Citation: Mikhalev, A.S.; Tynchenko, V.S.; Nelyub, V.A.; Lugovaya, N.M.; Baranov, V.A.; Kukartsev, V.V.; Sergienko, R.B.; Kurashkin, S.O. The Orb-Weaving Spider Algorithm for Training of Recurrent Neural Networks. *Symmetry* **2022**, *14*, 2036. https://doi.org/10.3390/sym14102036

Academic Editor: Jeng-Shyang Pan

Received: 15 July 2022
Accepted: 26 September 2022
Published: 29 September 2022

Abstract: The quality of operation of neural networks in solving application problems is determined by the success of the stage of their training. The task of learning neural networks is a complex optimization task. Traditional learning algorithms have a number of disadvantages, such as «sticking» in local minimums and a low convergence rate. Modern approaches are based on solving the problems of adjusting the weights of neural networks using metaheuristic algorithms. Therefore, the problem of selecting the optimal set of values of algorithm parameters is important for solving application problems with symmetry properties. This paper studies the application of a new metaheuristic optimization algorithm for weights adjustment—the algorithm of the spiders-cycle, developed by the authors of this article. The approbation of the proposed approach is carried out to adjust the weights of recurrent neural networks used to solve the time series forecasting problem on the example of three different datasets. The results are compared with the results of neural networks trained by the algorithm of the reverse propagation of the error, as well as three other metaheuristic algorithms: particle swarm optimization, bats, and differential evolution. As performance criteria for the comparison of algorithms of global optimization, in this work, descriptive statistics for metrics of the estimation of quality of predictive models, as well as the number of calculations of the target function, are used. The values of the *MSE* and *MAE* metrics on the studied datasets were obtained by adjusting the weights of the neural networks using the cycling spider algorithm at 1.32, 25.48, 8.34 and 0.38, 2.18, 1.36, respectively. Compared to the inverse error propagation algorithm, the cycling spider algorithm reduced the value of the error metrics. According to the results of the study, it is concluded that the developed algorithm showed high results and, in the assessment of performance, was not inferior to the existing algorithm.

Keywords: artificial intelligence; development and information communication technology; global optimization problem; meta-European algorithms; neural network weighting adjustment; recurrent neural networks; resource-use efficiency; spider-cycle algorithm

1. Introduction

Today artificial intelligence is a powerful tool for solving problems in various fields. Artificial Neural Networks (ANN) are one of the most promising areas for the development of artificial intelligence. They are used in the field of cybernetics, pattern recognition [1], prediction and forecasting [2], decision making [3], etc. Despite the fact that neural networks are used in a wide range of tasks, the task of increasing an ANN productivity increase is still urgent. One of the approaches to improve productivity is the optimal selection of the neural network hyperparameters, i.e., parameters that do not change during the neural network training (the number of hidden layers and neurons in each; the speed of the neural network training; the batch size for training; layer type; activation function; regularization parameters, etc.) [4].

The complexity of the neural networks hyperparameters settings is due to the fact that it is necessary to find a compromise between the high accuracy of solving the problem and the speed of the neural network training. When setting ANN models for solving specific problems and improving their quality during experiments, expert estimates and manual hyperparameters settings are used [5,6]. In the manual setting method, the researcher sets hyperparameters based on his own experience and, after training, evaluates the productivity.

Another approach to ANN hyperparameters settings is based on the use of automatic methods. One of the methods of automatic setting is based on the use of evolutionary algorithms [7,8].

Additionally, there are some approaches to solve the problem of adjusting the weights in the ANN model. One of these is to improve the quality of the designed ANN models considered in this work based on the development of new approaches to adjusting the weight coefficients based on the use of global optimization algorithms [9].

Currently, ANN are trained using various gradient algorithms and their modifications [10,11]. The algorithms have advantages, such as the purposefulness of the search and proven convergence, which make it possible to successfully apply them in solving the problems of ANN training. The algorithm converges to global (for the case of convex functions) or local minima (for the case of non-convex functions) [12]. However, gradient algorithms also have a number of disadvantages that make their application in specific tasks either problematic or completely impossible. As a rule, this is due to the fact that these algorithms are unable to get out of the local minima for the case of non-convex optimized functions [13,14]. Modern approaches to the problem of ANN parameter updating, as discussed in this paper, are based on the use of metaheuristic optimization algorithms [15] and do not require knowledge of the structure of the optimized function and calculation of the gradient. Another class is the non-iterative approach to learning neural networks [16–18].

Although metaheuristic algorithms vary widely, they do not always provide the required accuracy when setting neural network weights. This is due to the fact that optimization algorithms often focus on a certain class of problems, that is "there is no universal optimizer that could solve all classes of problems". Thus, the problem of adjusting neural network weights using optimization techniques involves the problem of choosing the most efficient algorithm, which involves experimenting with existing metaheuristic algorithms, improving them, as well as developing new approaches.

The main contributions of this article are as follows. First, the paper proposes a new algorithm for solving the global optimization problem based on the behavior of orb-weave spiders [19]. Second, the applicability of the developed algorithm is studied for solving the problem of setting the weights of neural networks.

To study the proposed algorithm, the task of setting the weights of the recurrent neural network used to predict the values of the time series was chosen, since it represents a complex multi-parameter optimization problem with many local optima.

The development and research of optimization algorithms are relevant, both for solving various applied problems, and for the science of artificial intelligence in general.

The practical value of the presented work for other researchers lies in the possibility of using proposed algorithm to solve a wide range of applied problems in various fields: industry, medicine, urban economy, etc.

In addition, the results are compared with the results obtained by other metaheuristic algorithms of global optimization [20–22]: particle swarm optimization [23], bats [24], and differential evolution [25–28]. The results obtained by applying the global optimization algorithm under consideration are also compared with the results of the classical algorithm of back error propagation [29]. Therefore, the rest of the article is organized as follows. Section two presents the task of time series forecasting and the literature review of metaheuristic algorithms. Section three describes the parameters of the studied optimization algorithms, the initial data, as well as the obtained results of adjustment of the neural network weights with the help of the studied algorithms. The program is available at https://github.com/digger32/Spider-Algorithm.git (accessed on 14 July 2022).

2. Materials and Methods

2.1. Problem Statement of the Time Series Forecasting

Recurrent neural networks are used to process time series events or sequential spatial circuits. The present study focuses on the time series forecasting problem, respectively; all the conclusions and results of experiments are valid for this application of recurrent ANN. The test of the effectiveness of the proposed approach in solving the problem of analysis of successive spatial chains is the subject of future research.

The problem of a time series forecasting has the following formulation. Let the values of the time series be given in discrete moments of time $T = \{1, 2, \dots, N\}$ [30,31]:

$$X = \{x(t), t \in T, x(t) \in R^n\} \quad (1)$$

where $x(t)$ is the value of the analyzed indicator registered at the t-th moment of time.

It is necessary, on the basis of the values of the analyzed indicator at the previous points in time $x(t)$, $x(t-1)$, $x(t-2)$, …, $x(t-k+1)$, $k \le N$, to predict (most accurately evaluate) the values of the analyzed indicator at moments $t + 1$, $t + 2$, …, $t + l$, i.e., and build a sequence of predicted values:

$$\hat{X} = \{\hat{x}(t+1), \hat{x}(t+2), \dots, \hat{x}(t+l)\} \quad (2)$$

To calculate the values of the time series at future points in time, it is required to determine a functional relationship $\hat{f}$ that reflects the relationship between the past and future values of this time:

$$\hat{x}(t+\tau) = \hat{f}\big(x(t-k+1), x(t-k+2), \dots, x(t+\tau-1)\big) \quad (3)$$

The presented functional dependence (3) is called the forecasting model.

Thus, the solution to the problem of a time series forecasting is to create a forecasting model that will meet the relevant criteria for the forecasting quality evaluation.

In this paper, we will consider the case of short-term forecasting, i.e., at $\tau = 1$.

2.2. Problem Statement of the Adjusting the Weight Coefficients of a Neural Network

The optimization approach to the problem of adjusting the neural network weights is to present this problem as a problem of minimizing the error function of the neural network. In general, this optimization problem is formulated as follows:

$$\min_W E(W) = \min_W \sum_{i=1}^{n} L\big(W, X^{(i)}, Y^{(i)}\big), \quad (4)$$

$$L\big(W, X^{(i)}, Y^{(i)}\big) = \big(h\big(W, X^{(i)}\big) - Y^{(i)}\big)^2. \quad (5)$$

where L—weighting error function, W is a matrix of neural network weights, $\left(X^{(i)}, Y^{(i)}\right)$—elements of the training sample, and $h\left(W, X^{(i)}\right)$—the value of the output signal of the neural network.

As a result of solving this problem, values of the weighting coefficients of the neural network should be found, where the neural network provides the minimum work error on the training sample.

2.3. Literature Review

The class of heuristic algorithms using patterns and principles borrowed from nature has proven itself especially well for solving global optimization problems. These include algorithms for evolutionary optimization [7] and swarm intelligence algorithms [32,33]. These algorithms belong to the class of population algorithms, which are based on modeling the collective behavior of self-organizing systems, the interacting elements which are called agents. Despite the lack of evidence on the convergence of these algorithms, they are often used in practice to solve complex optimization problems.

Evolutionary algorithms are based on a collective training process within an agent's population. The population is randomly initialized and then, at each step, the algorithm simulates the process of natural selection, when the stronger agents from the population outlive the weaker ones and produce the next generation [7].

A prominent representative of the evolutionary algorithm is the genetic algorithm (GA). It is a heuristic optimization algorithm originally based on the Darwinian principle of evolution through genetic selection. GA uses a very abstract version of evolutionary processes to develop solutions to given problems. Each GA handles a population of artificial chromosomes—these are strings in a finite alphabet (usually binary). Each chromosome represents a solution to a problem and has a fitness parameter—a real number, which is a measure of how well the solution fits a particular problem [34].

There is also an imperialist competition algorithm based on human social evolution modeling. However, unlike the traditionally used evolutionary algorithms, it is based on the social evolution of a person in society, and not on the biological evolution of species in nature [35].

Swarm intelligence optimization algorithms originate from nature. Unlike evolutionary algorithms, it is no longer required to create a new population at each step of the algorithm. Typical representatives of swarm intelligence algorithms are the ant algorithm (ACO) [36] and the particle swarm optimization algorithm (PSO) [23].

The use of metaheuristic algorithms in various applications is used in the scientific literature to adjust the weights of recurrent neural networks. The article [37,38] deals with the problem of adjusting neural network weights with long-term short-term memory for the task of predicting the vibration of the aircraft engine using the ant colony algorithm. Articles [39,40] discuss the application of the ant-lion algorithm for health and energy management problems. To solve practical problems in various applied fields (healthcare, ecology, mechanical engineering, energy, and stock market), the use of metaheuristic optimization algorithms are also found in the literature, such as: the gray wolf flock algorithm [40–42], whale optimization algorithm [43,44], cuckoo algorithm [45], etc. These articles show that that the application of metaheuristic optimization algorithms to adjust the weights of recurrent neural networks for each specific task leads to better performance and accurate results. In this regard, further study of the applicability of other metaheuristic optimization algorithms for adjusting the weights of recurrent neural networks remains relevant.

The algorithm proposed by the authors of the article is completely different from these algorithms in its biological roots, motivations, implementations, and search behavior. It is based on the peculiarities of the construction of the web and the competitive behavior of the orb weaving spiders. In this case, many spiders cannot be considered in the context of a "swarm", since each spider acts in its own interests. This, in turn, eases the rigidity of specimen selection, which is very difficult for other swarm algorithms.

For a comparative analysis, the following will be considered:

- Backpropagation algorithm (BP)—a classic algorithm for adjusting the weights of neural networks [46].
- Metaheuristic algorithms: particle swarm optimization algorithm (PSO) [26], differential evolution algorithm (DE) [20,22], bat algorithm (BI)—typical representatives of this class of algorithms, often used to solve a wide range of optimization problems [10,47].
- Orb-weaving spider algorithm (AA), proposed by the authors.

2.4. Description of Algorithms

2.4.1. Backpropagation Algorithm

Error backpropagation algorithm (BP) is a widely used neural network training algorithm for supervised training [29]. This algorithm is iterative and uses the principle of training "step by step" when the weights of the neurons of the network are corrected after submitting one training example to its input.

There are two network passes at each iteration—forward and backward. With a forward pass, the input vector propagates from the inputs of the network to its outputs and forms some output vector corresponding to the current (actual) state of the weight coefficients. The neural network error is then calculated as the difference between the actual and objective values. On the backward pass, this error propagates from the network output to its inputs, and the neuron weight coefficients are corrected.

The task of training a neural network is the task of minimizing the loss function in the space of weight coefficients. To solve this problem, gradient optimization algorithms are usually used; usually gradient descent or its modifications are used, for example, stochastic gradient descent.

2.4.2. Metaheuristic Algorithms

The differential evolution algorithm (DE) is a modification of the evolutionary optimization algorithm [25]. DE works by improving the collection of N possible solutions, which are being evaluated using the objective function f through the iterative process. At the first stage, a set of random vectors, called a generation, are initialized, which represent possible solutions to the optimization problem. Further, at each iteration, a new generation of vectors is generated randomly combining the vectors of the previous generation. After crossing, a selection operation is performed. If the resulting vector turns out to be better than the base vector (the value of the objective function has improved), then, in the new generation, the base vector is replaced with a trial one, otherwise the base vector remains in the new generation. In every era of the evolutionary process or at a given frequency, the best generation vector is determined in order to control the speed of finding the optimal solution.

The described stages of the differential evolution algorithm are repeated upon reaching a given number of iterations k.

The particle swarm optimization algorithm (PSO) is based on the movement of particles-agents in the search space and the evaluation of the attractiveness of the solutions found [23]. Each agent at some point in time is characterized by a vector of parameter values from the solution domain and the value of the function being optimized.

At each iteration of the algorithm, the agents change their position and speed depending on the previously found best solutions. Thus, as the study progresses, all agents begin to pull together in the area of the global solution. If an agent finds the best solution, then the values of the best positions for each agent of the entire system are updated. Further iterations are repeated until a certain stopping criterion is reached.

The bats algorithm (BI) is based on imitating the echolocation properties of bats [24]. As an agent approaches the local (or global) optimum, the volume of the emitted signal decreases, and its intensity increases. The approximation is determined by the current change in the value of the objective function.

At the first stage, a population of bats is initialized, each of which is characterized by position, speed, frequency, wavelength of the emitted pulse, and volume for searching of prey. The search of a solution is carried out taking into account the average loudness value of all bats at a certain time step. If the obtained solution is less than the best solution at the current step, then the volume is lowered, and the pulse propagation speed is increased.

2.4.3. Orb-Weaving Spider Algorithm

The orb-weaving spider algorithm is a heuristic competitive iterative random search algorithm, the main idea of which is to simulate the behavior of a diadem spider from the orb-weaving spider family [19].

Each iteration represents a day. During the day, the spider hunts, exploring the search area. At the end of the day, the spider looks for a new good place to build a web and destroys the old one, moving in the direction of the spider with the lowest (largest) function value.

If a spider enters the territory of another spider while moving, competition occurs. The stronger spider with the smallest (largest) function value at a given iteration wins and eats the spider with a greater (lower) function value.

An overview of the spider method is presented in Figure 1.

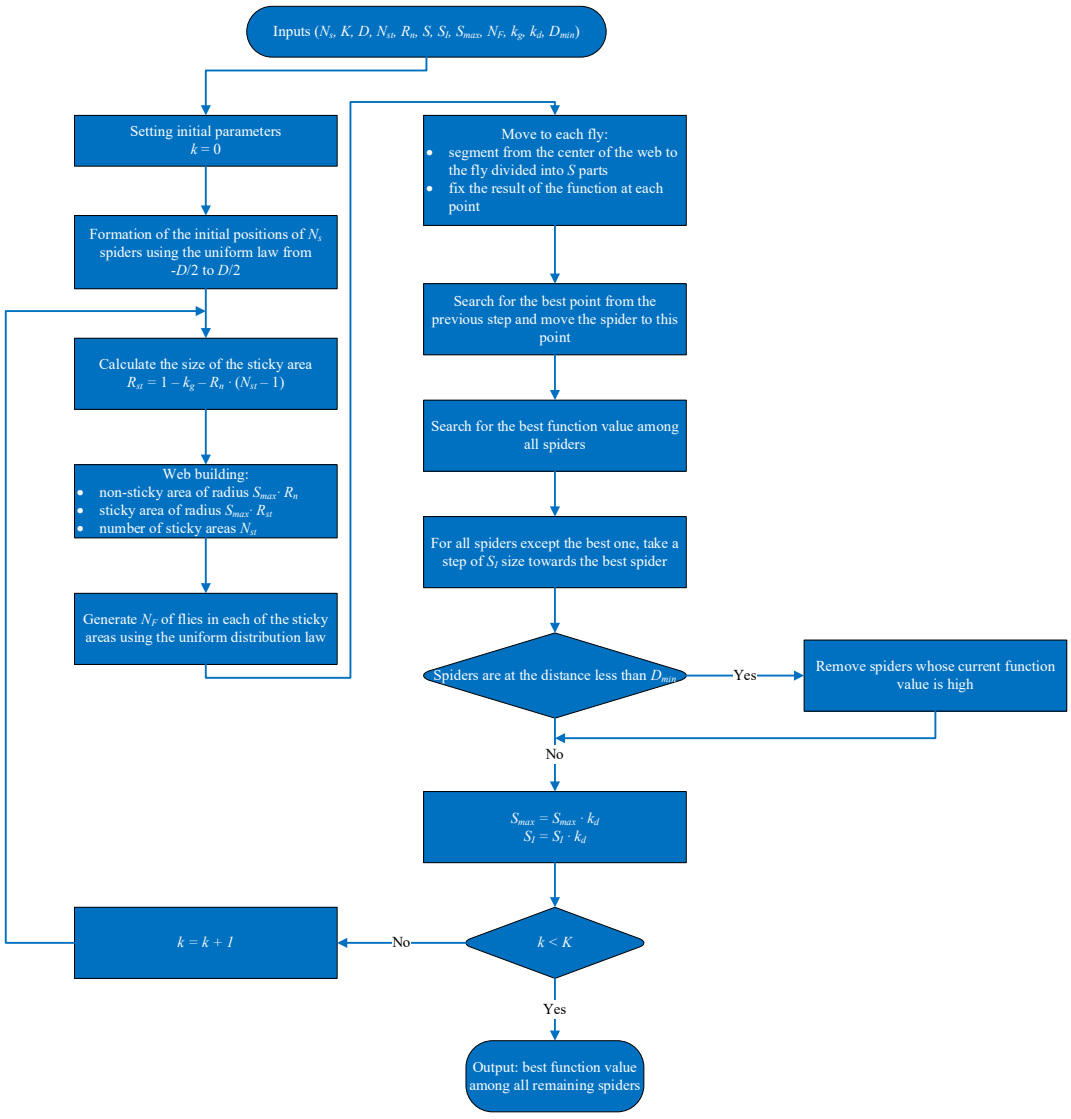

Figure 1. General scheme of the algorithm.

3. Experimental Study and Discussion

3.1. Algorithm Parameters

The efficiency of the algorithms in the search for the global minimum is caused by the setting of their respective parameters. Within the framework of this work, the selection of effective values of the parameters was carried out. The correct choice of such parameters has the greatest influence on the result of solving the optimization problem, namely, on achieving the minimum of the standard deviation function [48–50]. The results are shown in Tables 1–4.

Table 1. Particle swarm optimization algorithm.

Name	Description	Value
K	Maximum number of iterations	100
NP	Number of particles in a swarm	100
NI_{min}	Least number of neighbors	-
ω	Weight coefficient characterizing particle memory	0.5
α and β	Parameters used in calculating particle speed	0.5, 0.3

Table 2. Orb-weaving spider algorithm.

Name	Description	Value
K	Maximum number of iterations	100
N_s	Number of spiders	100
R_n	Radius of non-sticky area	0.05
N_{st}	Radius of sticky area	1
S	Number of steps	2
S_{max}	Maximum step size	0.3
S_I	Step after iteration	0
N_F	Number of flies	5
D_{min}	Minimum distance between spiders	0.001
k_g	Range coefficient of generation of flies	0.1
k_d	Coefficient for decreasing the variables S_{max} and S_I	0.95
K	Maximum number of iterations	25

Table 3. Bats algorithm.

Name	Description	Value
Num	Number of bats	100
$Iter$	Number of iterations	100
r	Signal pulsation frequency	0.8
a	Bat signal volume	0.8
Q_{min}	Minimum frequency	0
Q_{max}	Maximum frequency	2

Table 4. Differential evolution algorithm.

Name	Description	Value
K	Maximum number of iterations	100
Popsize	A multiplier to determine the total population	100
CR	Crossover probability	0.7
F	Differential weight	0.4

3.2. Data Description

To study the training algorithms, three datasets were taken from the Kaggle data science community website:

- Daily climate: Average daily temperature data from Delhi, India recorded over 3 years—from 1 January 2013 to 24 April 2017 [51,52].
- Advance retail sales: US Census Bureau dataset hosted on the Federal Reserve Economic Database [53].
- Solar radiation: Solar radiation dataset. Contains columns such as "wind direction", "wind speed", "humidity", and "temperature". Parameter "wind direction" is predicted [54].

Time series parameters are shown in Table 5.

Table 5. Series parameters.

Cell	Series Length	Mean Value	Min Value	Max Value	Standard Error
Daily climate	10,000	24.49	6.0	38.71	0.19
Advance retail sales	10,000	0.9	−29.5	23	0.55
Solar radiation	10,000	143.49	0.09	359.95	0.82

For time series forecasting, neural networks with long short-term memory (LSTM) have been designed and built.

The architecture of neural networks for solving each subtask was chosen individually. The number of LSTM layers varied from one to three, the number of neurons in each layer—from 50 to 100 with a step of 10. The setting up of other model hyperparameters was carried out using the random search technique with cross validation.

A description of the architectures of the obtained neural networks is presented in Table 6.

Table 6. Description of the architecture of the obtained neural networks.

Dataset	Number of Layers	Number of Neurons	Trainable Params	Accuracy
Daily climate	2	50	30,651	*MSE*: 2.31 *MAE*: 0.51
Advance retail sales	2	60	43,981	*MSE*: 29.62 *MAE*: 2.35
Solar radiation	3	60	73,021	*MSE*: 8.52 *MAE*: 2.25

3.3. Analysis and Comparison

To train the model, the sample was divided into training and test in a 3:1 ratio. For each of the algorithms, the following parameters were calculated: *MAE* and *MSE* [55]. To calculate the descriptive statistics for metrics of the estimation of quality of predictive models during the analysis, each algorithm was used to configure neural networks and was run m times:

- Mean squared error (*MSE*):

$$MSE = T^{-1}\sum_{t=1}^{T}\left(x(t) - \hat{x}(t)\right)^2 \quad (6)$$

where T—test part time series length.

- Mean absolute error (*MAE*):

$$MAE = T^{-1}\sum_{t=1}^{T}|x(t) - \hat{x}(t)| \quad (7)$$

- Median of the mean squared error:

$$\text{median}_{MSE} = median(MSE_1, \ldots, MSE_m) \quad (8)$$

- Median of the mean absolute error:

$$\text{median}_{MAE} = median(MAE_1, \ldots, MAE_m) \quad (9)$$

- The number of evaluations of the objective function values at trial points of the algorithm during its operation.

The calculation results are shown in Tables 7–9.

Table 7. Calculation results for dataset daily climate.

Indicator Name	Bats Algorithm	Particle Swarm Optimization Algorithm	Differential Evolution Algorithm	Spiders Algorithm
MSE	17.95	1.44	2.29	1.60
MAE	3.52	0.88	1.15	1.11
Median MSE	8.45	1.18	1.95	1.32
Median MAE	3.23	0.32	0.98	0.38
Calculations number	301,532	250,100	302,908	251,100

Table 8. Calculation results for dataset advance retail sales.

Indicator Name	Bats Algorithm	Particle Swarm Optimization Algorithm	Differential Evolution Algorithm	Spiders Algorithm
MSE	305.22	44.65	80.05	48.78
MAE	12.61	2.54	5.24	3.13
Median MSE	146.18	16.48	41.93	25.48
Median MAE	10.5	1.87	4.27	2.18
Calculations number	303,100	250,100	317,167	251,100

Table 9. Calculation results for dataset solar radiation.

Indicator Name	Bats Algorithm	Particle Swarm Optimization Algorithm	Differential Evolution Algorithm	Spiders Algorithm
MSE	12.23	8.74	10.53	9.17
MAE	2.71	1.82	1.92	1.82
Median MSE	8.66	8.14	9.38	8.34
Median MAE	2.30	1.64	1.27	1.36
Calculations number	303,100	250,100	218,971	251,100

Let us look at the results for each dataset. In order to compare metrics that indicate forecast errors, you need to know the maximum, minimum and average values for each dataset.

Based on the calculated data, we can conclude that the bats algorithm showed the worst results both in terms of the quality metrics of the obtained solution MSE and MAE, and in terms of the complexity of the computational process. On average for the three tasks, the MSE metric was 6.9 times higher, and the number of calculations was 20% higher compared to the best PSO algorithm.

The spider algorithm lags slightly behind the particle swarm optimization algorithm in terms of performance and has the same number of calculations, but is significantly ahead of the bats and differential evolution algorithms both in the number of calculations and in their quality:

1. According to the MSE metric, the spider algorithm outperforms the:
    - Bats algorithm in the daily climate, advance retail sales and solar radiation problems by 11.22, 6.26, and 1.25 times, respectively.
    - Differential evolution algorithm in the daily climate, advance retail sales, and solar radiation problems by 1.43, 1.64, and 1.15 times, respectively.
2. According to the MAE metric, the spider algorithm outperforms the:
    - Bats algorithm in the daily climate, advance retail sales, and solar radiation problems by 3.17, 4.03, and 1.49 times, respectively.
    - Differential evolution algorithm daily climate, advance retail sales, and solar radiation problems by 1.04, 1.67, and 1.05 times, respectively.

Figures 2–4 show the quality estimation of the algorithms relative to the best one (PSO) on the following problems: daily climate (Figure 2), advance retail sales (Figure 3), and solar radiation (Figure 4).

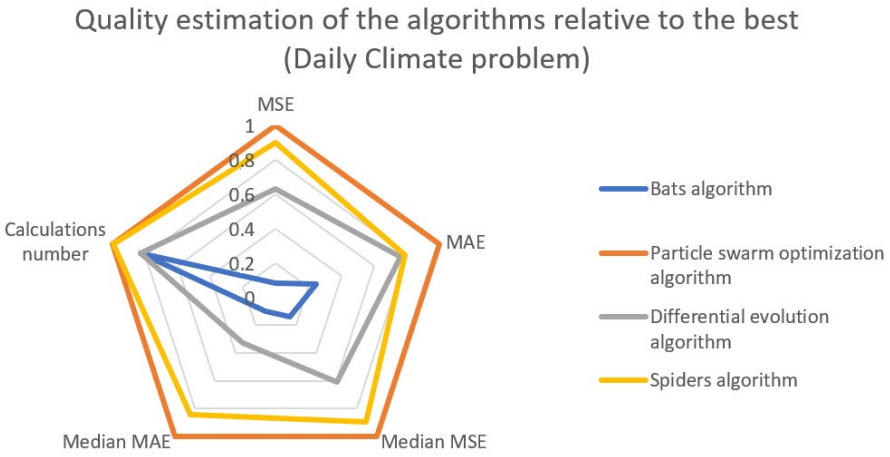

Figure 2. Quality estimation of the algorithms relative to the best one (PSO) on the daily climate problem.

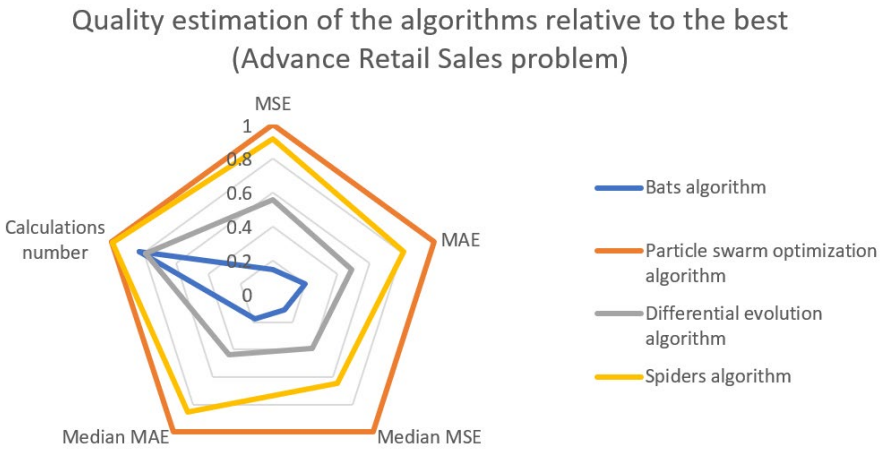

Figure 3. Quality estimation of the algorithms relative to the best one (PSO) on the advance retail sales problem.

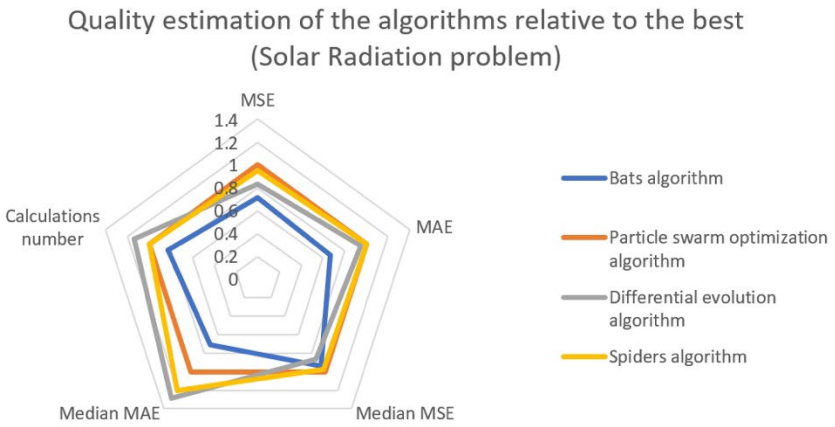

Figure 4. Quality estimation of the algorithms relative to the best one (PSO) on the solar radiation problem.

It can be seen that when solving the daily climate problem, the spider's algorithm showed significantly better results for all considered criteria compared to BI and DE, while being slightly inferior to PSO. When solving the advance retail sales problem, AA is even more superior to BI and DE, but also more inferior to PSO in terms of the median MSE criterion. In the solar radiation problem, all algorithms showed similar results, with AA outperforming PSO in the median MAE criterion.

Figures 5–7 show the results of forecast fragments for the compared algorithms on each dataset.

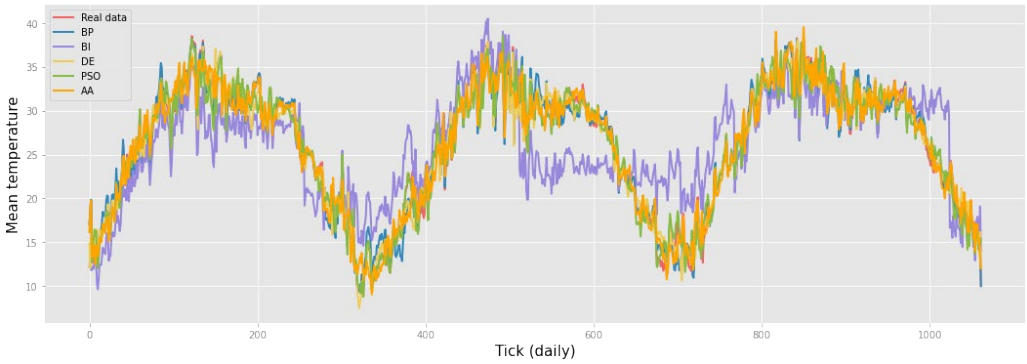

Figure 5. Fragment of the forecast result for the dataset daily climate, where the green lines is real data, red is the result of training by the orb-weaving spider algorithm, blue is by the differential evolution algorithm, orange is by the backpropagation algorithm, purple is by the bats algorithm, and yellow is by the particle swarm optimization algorithm.

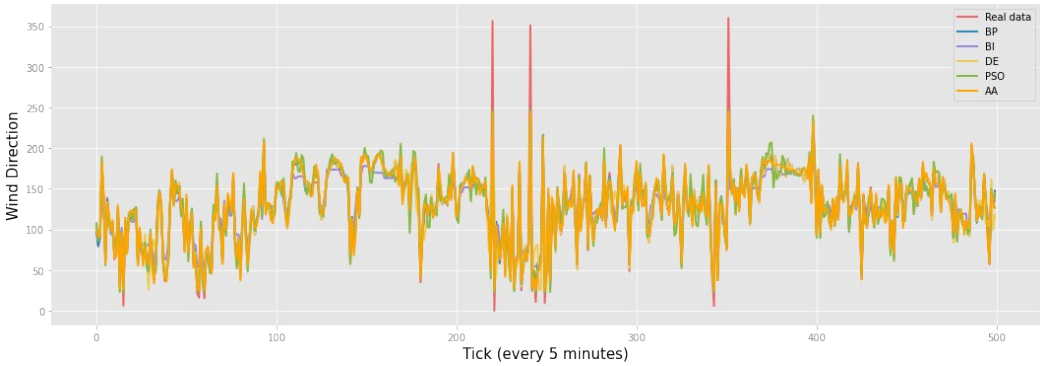

Figure 6. Fragment of the forecast result for dataset solar radiation, where the green line is real data, red is the result of training by the orb-weaving spider algorithm, blue is by the differential evolution algorithm, orange is by the backpropagation algorithm, purple is by the bats algorithm, and yellow is by the particle swarm optimization algorithm.

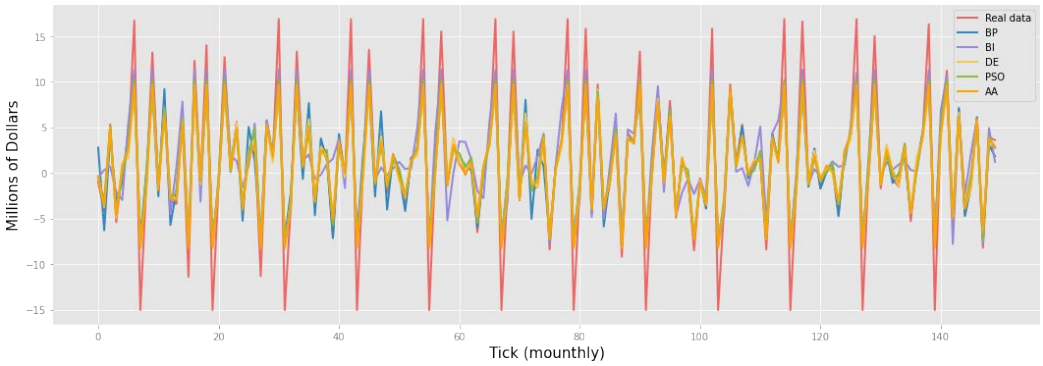

Figure 7. Fragment of the forecast result for dataset advance retail sales, where the green line is real data, red is the result of training by the orb-weaving spider algorithm, blue is by the differential evolution algorithm, orange is by the backpropagation algorithm, purple is by the bats algorithm, and yellow is by the particle swarm optimization algorithm.

Figure 5 shows that AA allows a good approximation of the seasonal fluctuations of the mean temperature, both in the annual cycle and in the seasonal warming at the end of each year.

In the solar radiation problem (Figure 6), the AA algorithm made it possible, not only to approximate the real data, but also to take into account the peaks of the wind direction on 218, 243, and 352 ticks.

In the advance retail sales task (Figure 7), the AA algorithm was allowed to train the recurrent neural network, which takes into account better negative peaks rather than positive.

Based on the presented fragments of forecasting results, we can conclude that the best results were shown by the PSO, AA, and BP algorithms. On datasets like daily climate and Advance Retail Sales, the PSO algorithm performed better than on datasets like solar radiation—AA. It is also worth noting that AA predicts the amplitude of the time series with a high percentage of probability.

4. Conclusions

This paper compares metaheuristic optimization algorithms using the example of setting up the weights of a recurrent neural network. The main attention is paid to the study of the properties of the new optimization algorithm for orb-weaving spiders.

In order to assess the effectiveness of AA, three datasets of different directions were selected and forecasting was performed. The comparison was happening with three classical optimization algorithms. The results showed the high efficiency of the proposed approach to solve the problem of recurrent neural networks training.

To date, there are many new metaheuristic optimization algorithms that are also applicable to the problem of tuning the weights of neural networks and, at the same time, can outperform the classical global optimization algorithms selected in this article. However, the comparison with classical algorithms, taking into account their existing standard software implementation, allows us to make an adequate comparative analysis. At the same time, as part of further research, it is planned to compare the effectiveness of the algorithm proposed in the article with other state-of-the-art metaheuristic algorithms on a wider and more representative set of tasks.

In addition, when solving machine learning problems using neural networks, it is important to tune, not only the weights of connections between neurons, but also the hyperparameters of neural networks, such as the number of layers, the number of neurons within each layer, and the type of activation function. In this article, to solve the problems under consideration, specific structures of neural networks that have become widespread have been chosen. However, a number of modern global optimization algorithms allow us to simultaneously adjust the weights and choose an optimal architecture of neural networks, which is a limitation of the presented study and a topic for future developments within the proposed approach.

Thus, from the point of view of expanding the capabilities of the proposed algorithm, the potential value lies in modifying this algorithm to solve, not only the problem of neural network training, but also architecture synthesis problems by extending the orb-weaving spider algorithm to solve discrete and mixed optimization problems [56–58].

In addition, one more future research question will be to study and improve the presented algorithm for setting up other types of neural networks, as well as to implement a multi-threaded version of the algorithm, which will significantly increase the speed of calculations.

Author Contributions: A.S.M., V.S.T. and V.V.K.; data curation, A.S.M., N.M.L., V.A.B., V.S.T., V.A.N. and R.B.S.; formal analysis, N.M.L., V.A.B., V.V.K., V.S.T., V.A.N. and S.O.K.; investigation, A.S.M., N.M.L., V.A.B., V.V.K., V.S.T., S.O.K., V.A.N. and R.B.S.; methodology, V.A.N., V.V.K. and A.S.M.; project administration, V.A.N. and V.S.T.; resources, V.A.B. and V.V.K.; software, A.S.M.; supervision, N.M.L., V.V.K. and V.S.T.; validation, A.S.M., N.M.L., V.A.B., V.A.N., R.B.S. and S.O.K.; visualization, V.S.T., V.V.K., R.B.S. and A.S.M.; writing—original draft, A.S.M., N.M.L., V.A.B., V.S.T., V.A.N., R.B.S., V.V.K. and S.O.K.; writing—review & editing, N.M.L., V.A.B., V.S.T., V.A.N., R.B.S., V.V.K. and S.O.K. All authors have read and agreed to the published version of the manuscript.

Funding: The studies were carried out within the program of the Russian Federation of strategic academic leadership "Priority-2030" aimed at supporting the development programs of educational

institutions of higher education, the scientific project PRIOR/SN/NU/22/SP5/16 "Building intelligent networks, determining their structure and architecture, operation parameters in order to increase productivity systems and the bandwidth of data transmission channels using trusted artificial intelligence technologies that provide self-learning, self-adaptation, and optimal reconfiguration of intelligent systems for processing large heterogeneous data".

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Basu, J.K.; Bhattacharyya, D.; Kim, T.-H. Use of artificial neural network in pattern recognition. *Int. J. Softw. Eng. Its Appl.* **2010**, *4*.
2. Hewamalage, H.; Bergmeir, C.; Bandara, K. Recurrent Neural Networks for Time Series Forecasting: Current status and future directions, *Int. J. Forecast.* **2021**, *37*, 388–427.
3. Kochenderfer, M.J.; Wheeler, T.A.; Wray, K.H. *Algorithms for Decision Making*; The MIT Press: Cambridge, MA, USA, 2022.
4. Nur, A.S.; Radzi, N.H.M.; Ibrahim, A.O. Artificial Neural Network Weight Optimization. *Telkomnika* **2014**, *10*, 11591.
5. Bergstra, J.; Bardenet, R.; Bengio, Y.; Kegl, B. Algorithms for hyperparameter optimization. In Proceedings of the 24th International Conference on Neural Information Processing Systems (NIPS'11), Granada, Spain, 12–15 December 2011; Curran Associates Inc.: New York, NY, USA, 2011; pp. 2546–2554.
6. Hovden, I.T. Optimizing Artificial Neural Network Hyperparameters and Architecture, 2019. https://www.mn.uio.no/fysikk/english/people/aca/ivarth/works/in9400_nn_hpo_nas_hovden_r2.pdf. (accessed on 25 September 2022).
7. Maier, H.R.; Razavi, S.; Kapelan, Z.; Matott, L.S.; Kasprzyk, J.; Tolson, B.A. Introductory overview: Optimization using evolutionary algorithms and other metaheuristics. *Environ. Model. Softw.* **2019**, *114*, 195–213.
8. Srivastava, S.; Sahana, S.K. A Survey on Traffic Optimization Problem Using Biologically Inspired Techniques. *Nat. Comput.* **2020**, *19*, 647–661. https://doi.org/10.1007/S11047-019-09731-Z/FIGURES/7.
9. Hamm, L.; Wade Brorsen, B.; Hagan, M.T. Global optimization of neural network weights. In Proceedings of the 2002 International Joint Conference on Neural Networks (IJCNN'02), Honolulu, HI, USA, 12–17 May 2002; Volume 2, pp. 1228–1233.
10. Jiang, M.; Liu, W.; Xu, W.; Chen, W. Improved Multiobjective Bat Algorithm for the Credibilistic Multiperiod Mean-VaR Portfolio Optimization Problem. *Soft Comput.* **2021**, *25*, 6445–6467. https://doi.org/10.1007/S00500-021-05638-Z/TABLES/14.
11. Ruder, S. An overview of gradient descent optimization. *arXiv* **2016**, arXiv:1609.04747.
12. Lee, J.D.; Simchowitz, M.; Jordan, M.I.; Recht, B. Gradient descent only converges to minimizers. In Proceedings of the Conference on Learning Theory, New York, NY, USA, 23–26 June 2016; pp. 1246–1257.
13. Du, S.S.; Jin, C.; Lee, J.D.; Jordan, M.I.; Singh, A.; Poczos, B. Gradient descent can take exponential time to escape saddle points. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1067–1077.
14. Zhu, Z., Soudry, D., Eldar, Y.C., Wakin, M.B.. The Global Optimization Geometry of Shallow Linear Neural Networks. *J. Math. Imaging Vis.* **2020**, *62*, 1067–1077.
15. Devikanniga, D.; Vetrivel, K.; Badrinath, N. Review of Meta-Heuristic Optimization based Artificial Neural Networks and its Applications. *J. Phys. Conf. Ser.* **2019**, *1362*, 012074.
16. Wang, X.; Cao, W. Non-iterative approaches in training feed-forward neural networks and their applications. *Soft Comput.* **2018**, *22*, 3473–3476.
17. Tkachenko, R.; Izonin, I.; Vitynskyi, P.; Lotoshynska, N.; Pavlyuk, O. Development of the Non-Iterative Supervised Learning Predictor Based on the Ito Decomposition and SGTM Neural-Like Structure for Managing Medical Insurance Costs. *Data* **2018**, *3*, 46. https://doi.org/10.3390/data3040046.
18. Izonin, I.; Tkachenko, R.; Kryvinska, N.; Tkachenko, P.; Gregus, M. Multiple Linear Regression Based on Coefficients Identification Using Non-iterative SGTM Neural-like Structure. In Proceedings of the International Work-Conference on Artificial Neural Networks, Gran Canaria, Spain, 12–14 June 2019; pp. 467–479. https://doi.org/10.1007/978-3-030-20521-8_39.
19. Baranov, V.A.; Lugovaya, N.M.; Mikhalev, A.S.; Kudymov, V.I.; Strekaleva, T.V. The algorithm of overall optimization based on the principles of intraspecific competition of orb-web spiders. *IOP Conf. Ser. Mater. Sci. Eng.* **2020**, *734*, 012141.
20. Jeya, R.; Venkatakrishnan, G.R.; Rengara, R.; Anand, S.S.; Bharath Raj, N.; Ramanathan, G. Evolutionary Optimization Algorithms—A Review. *J. Adv. Res. Dyn. Control. Syst.* **2018**, *10*, 1112–1122.
21. Houssein, E.H.; Mahdy, M.A.; Eldin, M.G.; Shebl, D.; Mohamed, W.M.; Abdel-Aty, M. Optimizing Quantum Cloning Circuit Parameters Based on Adaptive Guided Differential Evolution Algorithm. *J. Adv. Res.* **2021**, *29*, 147–157. https://doi.org/10.1016/J.JARE.2020.10.001.
22. Vala, T.M.; Rajput, V.N.; Geem, Z.W.; Pandya, K.S.; Vora, S.C. Revisiting the Performance of Evolutionary Algorithms. *Expert Syst. Appl.* **2021**, *175*, 114819. https://doi.org/10.1016/J.ESWA.2021.114819.

23. Jain, N.K.; Nangia, U.; Jain, J. A Review of Particle Swarm Optimization. *J. Inst. Eng. Ser. B* **2018**, *99*, 407–411.
24. Al-Betar, M.A.; Awadallah, M.A.; Faris, H.; Yang, X.-S.; Tajudin Khader, A.; Alomari, O.A. Bat-inspired algorithms with natural selection mechanisms for global optimization. *Neurocomputing* **2018**, *273*, 448–465.
25. Bilal, Pant, M.; Zaheer, H.; Garcia-Hernandez, L.; Abraham, A. Differential Evolution: A review of more than two decades of research. *Eng. Appl. Artif. Intell.* **2020**, *90*, 103479.
26. Jahandideh-Tehrani, M.; Bozorg-Haddad, O.; Loáiciga, H.A. Application of Particle Swarm Optimization to Water Management: An Introduction and Overview. *Environ. Monit. Assess.* **2020**, *192*, 281. https://doi.org/10.1007/S10661-020-8228-Z/TABLES/1.
27. Aksenova, M.M.; Bondarenko, V.L.; Kupriyanov, M.Y. An Automated System for Control of Refrigerant Parameters in a Fractionating Plant. *Chem. Pet. Eng.* **2020**, *56*, 433–439. https://doi.org/10.1063/5.0035767.
28. Shen, K.; Liu, R.; Guo, R.; Neusypin, K.A.; Proletarsky, A.V. Technology of error compensation in navigation systems based on nonlinear Kalman filter. *Guofang Keji Daxue Xuebao* **2017**, *39*, 84–90. https://doi.org/10.11887/j.cn.201702012.
29. Aggarwal, C.C. *Neural Networks and Deep Learning: A Textbook*; Springer International: Cham, Switzerland, 2018.
30. Weise, T. *Global Optimization Algorithms-Theory and Applications*, 3rd ed.; University of Science and Technology of China: Hefei, China, 2011; p. 1217.
31. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, UK, 2016.
32. Bansal, J.C.; Singh, P.K.; Pal, N.R. *Evolutionary and Swarm Intelligence Algorithms. Studies in Computational Intelligence*; Springer: Cham, Switzerland, 2019.
33. Nayyar, A.; Le, D.-N.; Nguyen N.G. *Advances in Swarm Intelligence for Optimizing Problems in Computer Science*; CRC Press: New York, NY, USA, 2019.
34. Katoch, S.; Chauhan, S.S.; Kumar, V. A review on genetic algorithm: Past, present, and future. *Multimed. Tools Appl.* **2021**, *80*, 8091–8126.
35. Maheri, M.R.; Talezadeh, M. An Enhanced Imperialist Competitive Algorithm for optimum design of skeletal structures. *Swarm Evol. Comput.* **2018**, *40*, 24–36.
36. Dorigo, M.; Stutzle, T. Ant Colony Optimization: Overview and Recent Advances In Handbook of Metaheuristics; Springer: Cham, Switzerland, 2019; pp. 311–351.
37. Elsaid, A.; Higgins, J.; Wild, B.; Desell, T. Using Ant Colony Optimization to Optimize Long Short-Term Memory Recurrent Neural Networks. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 18), Kyoto, Japan, 15–19 July 2018; pp. 13–20.
38. Elsaid, A.; El Jamiy, F.; Higgins, J.; Wild, B.; Desell, T. Optimizing long short-term memory recurrent neural networks using ant colony optimization to predict turbine engine vibration. *Appl. Soft Comput. J.* **2018**, *73*, 969–991.
39. Roy, K.; Mandal, K.K.; Mandal, A.C. Ant-Lion Optimizer algorithm and recurrent neural network for energy management of micro grid connected system, *Energy* **2019**, *167*, 402–416.
40. Kose, U. An ant-lion optimizer-trained artificial neural network system for chaotic electroencephalogram (EEG) prediction. *Appl. Sci.* **2018**, *8*, 1613.
41. Hu, H.; Wang, H.; Bai, Y.; Liu, M. Determination of endometrial carcinoma with gene expression based on optimized Elman neural network. *Appl. Math. Comput.* **2019**, 341, 204–214.
42. Xing, Y.; Yue, J.; Chen, C.; Xiang, Y.; Chen, Y.; Shi, M. A deep belief network combined with modified greywolf optimization algorithm for PM2.5 concentration prediction. *Appl. Sci.* **2019**, *9*, 3765.
43. Wang, J.; Wang, S.; Yang, W. A novel non-linear combination system for short-term wind speed forecast. *Renew. Energy* **2019**, *143*, 1172–1192.
44. Wang, J.; Du, P.; Niu, T.; Yang, W. A novel hybrid system based on a new proposed algorithm—Multi-Objective Whale Optimization Algorithm for wind speed forecasting. *Appl. Energy* **2017**, *208*, 344–360.
45. Nawi, N.M.; Khan, A.; Rehman, M.Z.; Chiroma, H.; Herawan, T. Weight Optimization in Recurrent Neural Networks with Hybrid Metaheuristic Cuckoo Search Techniques for Data Classification. *Math. Probl. Eng.* **2015**, *2015*, 868375.
46. Yang, S.; Chen, D.; Li, S.; Wang, W. Carbon price forecasting based on modified ensemble empirical mode decomposition and long short-term memory optimized by improved whale optimization algorithm. *Sci. Total Environ.* **2020**, *716*, 137117.
47. Liao, R.; Xiong Yu Fetaya, E.; Zhang, L.; Yoon, K.; Pitkow, X.; Urtasun, R.; Zemel, R. Reviving and Improving Recurrent Back-Propagation. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; Volume 80, pp. 3082–3091.
48. Kennedy, J.; Eberhart, R.C.; Shi, Y. *Swarm Intelligence—Russell, C. Eberhart, Yuhui Shi, James Kennedy—Google Docs*; Academic Press: San Diego, CA, USA, 2001.
49. Georgioudakis, M.; Plevris, V. A Comparative Study of Differential Evolution Variants in Constrained Structural Optimization. *Front. Built Environ.* **2020**, *6*, 102. https://doi.org/10.3389/FBUIL.2020.00102/XML/NLM.
50. Menassel, R.; Gaba, I.; Titi, K. Introducing BAT Inspired Algorithm to Improve Fractal Image Compression. *Int. J. Comput. Appl.* **2019**, *42*, 697–704. https://doi.org/10.1080/1206212X.2019.1638631.
51. Mousavirad, S.J.; Schaefer, G.; Jalali, S.M.J.; Korovin, I. A benchmark of recent population-based metaheuristic algorithms for multi-layer neural network training. In Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20), Cancún, Mexico, 8–12 July 2020; pp. 1402–1408.

52. Daily Climate Time Series Data. Available online: https://www.kaggle.com/sumanthvrao/daily-climate-time-series-data (accessed on 25 September 2022).
53. Advance Retail Sales Time Series Collection. Available online: https://www.kaggle.com/census/advance-retail-sales-time-series-collection/version/41 (accessed on 25 September 2022).
54. Solar Radiation Prediction. Available online: https://www.kaggle.com/dronio/SolarEnergy (accessed on 25 September 2022).
55. Wang, W.; Lu, Y. Analysis of the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE) in Assessing Rounding Model. *IOP Conf. Ser. Mater. Sci. Eng.* **2018**, *324*, 012049.
56. Wang, F.; Zhang, H.; Zhou, A. A Particle Swarm Optimization Algorithm for Mixed-Variable Optimization Problems. *Swarm Evol. Comput.* **2020**, *60*, 100808.
57. Al-Madi, N.; Faris, H.; Mirjalili, S. Binary multi-verse optimization algorithm for global optimization and discrete problems. *Int. J. Mach. Learn. Cybern.* **2019**, *10*, 3445–3465.
58. Chudakov, O.; Gorelov, V.; Padalkin, B. Mathematical Modeling of a Linear Motion on a Deformable Bearing Surface of a Saddle-Type Road Train with Active Semi-Trailer Element. *IOP Conf. Ser. Mater. Sci. Eng.* **2020**, *820*, 012009. https://doi.org/10.1088/1757-899X/820/1/012009.