energies

MDPI

Article

A Short-Term Load Forecasting Model Based on Crisscross Grey Wolf Optimizer and Dual-Stage Attention Mechanism

Renxi Gong [1,2,*] and Xianglong Li [1]

[1] School of Electrical Engineering, Guangxi University, Nanning 530004, China
[2] School of Traffic &Transportation, Nanning University, Nanning 530200, China
* Correspondence: rxgong@gxu.edu.cn

Abstract: Accurate short-term load forecasting is of great significance to the safe and stable operation of power systems and the development of the power market. Most existing studies apply deep learning models to make predictions considering only one feature or temporal relationship in load time series. Therefore, to obtain an accurate and reliable prediction result, a hybrid prediction model combining a dual-stage attention mechanism (DA), crisscross grey wolf optimizer (CS-GWO) and bidirectional gated recurrent unit (BiGRU) is proposed in this paper. DA is introduced on the input side of the model to improve the sensitivity of the model to key features and information at key time points simultaneously. CS-GWO is formed by combining the horizontal and vertical crossover operators, to enhance the global search ability and the diversity of the population of GWO. Meanwhile, BiGRU is optimized by CS-GWO to accelerate the convergence of the model. Finally, a collected load dataset, four evaluation metrics and parametric and non-parametric testing manners are used to evaluate the proposed CS-GWO-DA-BiGRU short-term load prediction model. The experimental results show that the RMSE, MAE and SMAPE are reduced respectively by 3.86%, 1.37% and 0.30% of those of the second-best performing CSO-DA-BiGRU model, which demonstrates that the proposed model can better fit the load data and achieve better prediction results.

Keywords: short-term load prediction; dual-stage attention mechanism; crisscross grey wolf optimizer

check for updates

Citation: Gong, R.; Li, X. A Short-Term Load Forecasting Model Based on Crisscross Grey Wolf Optimizer and Dual-Stage Attention Mechanism. *Energies* **2023**, *16*, 2878. https://doi.org/10.3390/en16062878

Academic Editors: Ali Ahmadian and Ali Almansoori

Received: 9 February 2023
Revised: 15 March 2023
Accepted: 17 March 2023
Published: 21 March 2023

1. Introduction

Electric load forecasting plays an important role in the modernization of power system management and has become the research focus of current power enterprises [1]. It can be divided into long-term, medium-term and short-term forecasting according to its different purposes [2]. Among them, short-term load forecasting can ensure the safe and stable operation of the power system and improve social benefits [3]. Furthermore, it can accelerate the development of the power market and improve economic benefits [4]. Therefore, it is of great significance to design an efficient and accurate short-term load forecasting method.

The early short-term load methods mainly use are the exponential smoothing method [5] and hidden Markov model [6], but the ability of these methods to extract nonlinear characteristics of load is weak [7]. With the rapid increase in the installation of smart meters [8] and the development of artificial intelligence technology [9], short-term load forecasting based on big data analysis has become a current research hotspot, such as the BP neural network [10], extreme learning machine [11], support vector machine [12], etc. In addition, in order to avoid falling into local minima, some scholars use swarm intelligence optimization algorithms to optimize the artificial intelligence model. For example, Niu and Dai [13] proposed a short-term load forecasting model based on modified particle swarm optimization, in which the parameters of least squares supporting a vector machine are optimized. The experimental results show that the regression accuracy and generalization ability of the model have been improved by the proposed algorithm. To address the problem of the ease in which long short-term memory neural networks fall into local minima, a whale optimization algorithm (WOA) is used to optimize the network [14]. Li et al. [15] use grey wolf optimization (GWO) to optimize the parameters of every single kernel in an extreme learning machine to improve its forecasting ability. However, GWO quickly falls into the optima trap or fails to find the global optimal solution [16].

Moreover, the above-mentioned shallow learning method is applicable in the scenario where only the historical load is used for forecasting. If it is necessary to extract deep hidden features in massive load data, a deep learning model is needed. For example, Khan et al. [17] use a convolution neural network to extract the coupling relationship of the input features. Muzaffar and Afshari [18] use a long short-term memory (LSTM) neural network to learn the temporal correlation contained in the load time series data. In addition, a gated recurrent unit (GRU) with simpler structures is also used for short-term load prediction [19], with a high-efficiency ability of feature extraction. To extract the implicit coupling relationship between features and temporal dependency in load time series, the combination of CNN and LSTM [20], or its improved variants (e.g., CNN-GRU [21], GRU-TCN [22] and RCNN-ML-LSTM [23]) are used in load prediction. However, the above deep learning models have the problems of gradient disappearance and gradient explosion [24]. Therefore, it is of great significance to avoid these problems and accelerate the convergence speed of the model, so as to improve the accuracy of load forecasting.

The above-mentioned short-term load forecasting models based on artificial intelligence techniques do not consider the importance of input features, making important features disappear with the increase of step size [25]. Feature selection is the commonly used technique to select out the most appropriate input features in forecasting problems [26]. Kong et al. [27] utilize principal component analysis to determine the major factors affecting wind speed, reducing the dimension of relative features and improving the generation of model. Li et al. [28] develop a feature selection method to choose competitive input features. The above feature selection methods reduce the number of input features only once before prediction by simple correlation analysis, which causes the potentially important input variable be discarded [29]. To address this problem, an attention mechanism is proposed [30], with the advantage of making the model handle the dependency of long time series more easily. For example, Wang et al. [31] proposed a short-term load forecasting model based on a feature attention mechanism (FA), in which the effective characteristics of the input variables are highlighted, leading to improved prediction accuracy. In Ref. [32], a temporal attention mechanism (TA) is applied in short-term load prediction to capture the high-impact time steps of load sequence, so as to further reduce the prediction errors. However, there are few research studies that focus on combining FA with TA to propose a multi-stage attention mechanism, capturing the feature and temporal relationship in load time series.

In view of the shortcomings of existing forecasting models, this paper proposes a short-term load forecasting model (DA-CS-GWO-BiGRU) based on a dual-stage attention mechanism (DA) and crisscross grey wolf optimizer algorithm (CS-GWO). The contributions of this paper are presented as follows:

- Combining the advantages of a feature and temporal attention mechanism, a dual-stage attention mechanism (DA) is introduced in this paper. DA is utilized at the input side of the forecasting model to comprehensively capture the correlation relationship between various variables and temporal dependency in the load time series.
- To address the deficiency of GWO, a novel crisscross grey wolf optimizer algorithm is firstly applied in a short-term load forecasting problem. By introducing horizontal and vertical crossover operators, the global search ability and community diversity of CS-GWO are improved.
- The proposed DA-CS-GWO-BiGRU model is verified by using the real load data set collected in a certain area. The experimental results show that the proposed model has higher forecasting accuracy than other comparison models, and has good application prospects.

The organization of the remainder of this paper is as follows. Section 2 introduces the basic principle of the deep learning model involved in this paper. Section 3 presents the methodology of the proposed DA-CS-GWO-BiGRU short-term load forecasting model. Section 4 introduces the metrics of evaluating predictions. Section 5 focuses on the details of the experiments, and the results are analyzed and discussed. Section 6 points out the limitations of and indicates the subsequent work that could follow this paper. Finally, Section 7 summarizes this paper.

2. Principle of Deep Learning Model

2.1. BiGRU Neural Network

A recurrent neural network (RNN) can only remember short-term dependencies of a time series and is often accompanied by the problem of gradient explosion or disappearance in the training process [33], leading to its limited use in practice. By modifying the calculation method of the hidden state of RNN, GRU and LSTM can effectively strengthen the long-term dependence of time series [19]. The network structure of GRU is shown in Figure 1.

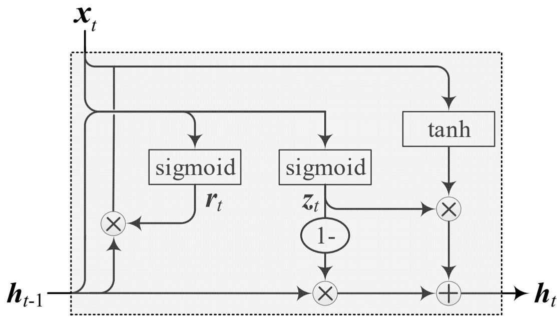

Figure 1. The network structure of GRU.

It can be seen from Figure 1 that the GRU network calculates the combination degree of the current input and the previous status information by the reset gate $\boldsymbol{r}_t$. The calculation process is shown in Equation (1).

$$\boldsymbol{r}_t = \sigma(\boldsymbol{x}_t W_r + \boldsymbol{h}_{t-1} U_r + b_r) \tag{1}$$

where $\boldsymbol{x}_t$ is the input data in t-th time step, $\boldsymbol{h}_{t-1}$ is the output of the previous time step, W_r and U_r are the weight metrices of the reset gate, b_r is the bias of the reset gate and σ is the sigmoid activation function.

In addition, the GRU network controls the retention of the previous state information $\boldsymbol{h}_{t-1}$ in the current state by update gate z_t, and its calculation process is shown in Equation (2).

$$z_t = \sigma(\boldsymbol{x}_t W_z + \boldsymbol{h}_{t-1} U_z + b_z) \tag{2}$$

where W_z and U_z are the weight metrices of the update gate and b_z is the bias of the update gate.

Next, GRU obtains candidate hidden states through the reset gate based on the updating mechanism of RNN [34], as shown in Equation (3).

$$\widetilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{x}_t W_h + (\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) U_h + b_h) \tag{3}$$

where W_h and U_h are the weight metrices of candidate output and b_h is the bias of the candidate output.

Finally, GRU obtains new hidden state $\boldsymbol{h}_t$ by considering the previous hidden state $\boldsymbol{h}_{t-1}$ and candidate hidden state $\widetilde{\boldsymbol{h}}_t$ as well as z_t. The calculation process is shown in Equation (4).

$$\boldsymbol{h}_t = z_t \odot \boldsymbol{h}_{t-1} + (1 - z_t) \odot \widetilde{\boldsymbol{h}}_{t-1} \tag{4}$$

It can be seen from Equations (1)–(4) that GRU only considers the influence factors on the current time, and lacks consideration of future influence factors [35]. Meanwhile, a recently proposed neural network named bidirectional gated recurrent unit (BiGRU) can effectively make up for the deficiency of GRU. BiGRU can fully excavate the influence relationship hidden in the time series before and after through its unique forward and backward propagation network structure [36]. The network structure of BiGRU is shown in Figure 2, and its calculation process is shown in Equations (5)–(7).

$$\overrightarrow{\boldsymbol{h}}_t = G(\boldsymbol{x}_t, \overrightarrow{\boldsymbol{h}}_{t-1}) \tag{5}$$

$$\overleftarrow{\boldsymbol{h}}_t = G(\boldsymbol{x}_t, \overleftarrow{\boldsymbol{h}}_{t-1}) \tag{6}$$

$$\boldsymbol{h}_t = \overrightarrow{\boldsymbol{w}}_t \overrightarrow{\boldsymbol{c}}_t + \overleftarrow{\boldsymbol{w}}_t \overleftarrow{\boldsymbol{c}}_t + \boldsymbol{b}_t \tag{7}$$

where $\overrightarrow{\boldsymbol{h}}_t$ is the state information of forward propagation, $\overleftarrow{\boldsymbol{h}}_t$ is the state information of backward propagation, $\overrightarrow{\boldsymbol{w}}_t$ is the weight metrices of the hidden layer in forward propagation, $\overleftarrow{\boldsymbol{w}}_t$ is the weight metrices of the hidden layer in backward propagation, $\boldsymbol{b}_t$ is the bias of the hidden layer and $G(\cdot)$ is the calculation process of GRU as shown in Equations (1)–(4).

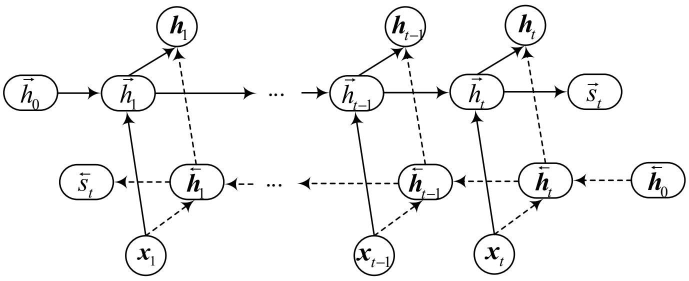

Figure 2. The network structure of BiGRU.

As can be seen from Figure 2, BiGRU can exploit information both from the past and the future. Therefore, this paper uses BiGRU for timing analysis.

2.2. Attention Mechanism

By simulating human visual behavior, the attention mechanism adaptively assigns different attention weights to the input features of the model to highlight the more critical influence factors [37], helping the model predict better.

The attention mechanism is mainly composed of three parts, namely, attention weight calculation, weight normalization and intermediate semantic vector calculation. Firstly, the attention weight $\boldsymbol{e}$ of different features in the model input $\boldsymbol{x}$ or at t-th time step is calculated by using a multi-layer perceptron or neural network. Then, in order to meet the requirement that the sum of attention weights is 1, the attention weight $\boldsymbol{e}$ is normalized to find $\boldsymbol{\alpha}$. Finally, the intermediate semantic vector can be obtained by considering $\boldsymbol{x}$ and $\boldsymbol{\alpha}$ as shown in Equation (8).

$$\boldsymbol{c} = \boldsymbol{\alpha}\boldsymbol{x} \tag{8}$$

Therefore, this paper expects to use the attention mechanism to capture the coupling relationship between each feature and the impact of information both from the past and the future on the forecasted load value.

3. DA-CS-GWO-BiGRU Short-Term Load Forecasting Model

3.1. Mathematical Model

Set $\boldsymbol{L}^{(i)} = (L(24 \cdot i - 23), \ldots, L(24 \cdot i))$ as the electric load time series of the previous day, $\boldsymbol{t}^{(i)} = (t_{\max}(i), t_{\max}(i+1), t_{\min}(i), t_{\min}(i+1))$ as the highest and lowest temperature of the previous day and the current day, $\boldsymbol{r}^{(i)} = (r(i), r(i+1))$ as the rainfall of the previous day and the current day and $\boldsymbol{d}^{(i)} = (d(i), d(i+1))$ as the weather day type of the previous day and the current day. The short-term load forecasting problem can be regarded as using the load information of the previous day $\boldsymbol{L}^{(i)}$, and combining its relevant characteristics $\boldsymbol{t}^{(i)}$, $\boldsymbol{r}^{(i)}$ and $\boldsymbol{d}^{(i)}$ to make a prediction of the electric load values in the current day. Let the function map of the model be F_θ, and the prediction process is shown in Equation (9).

$$\hat{\boldsymbol{Y}}^{(i)} = F_\theta(\boldsymbol{X}^{(i)}) \tag{9}$$

where $\boldsymbol{X}^{(i)} = \left[\boldsymbol{L}^{(i)}, \boldsymbol{t}^{(i)}, \boldsymbol{r}^{(i)}, \boldsymbol{d}^{(i)}\right] = \left(x_1^{(i)}, x_2^{(i)}, \ldots, x_T^{(i)}\right) \in \mathbb{R}^T$, and $\hat{\boldsymbol{Y}}^{(i)}$ represents the predicted load values of the current day.

3.2. Dual-Stage Attention Mechanism

As shown in Section 3.1, the prediction model takes historical load time series $\boldsymbol{L}^{(i)}$, temperature $\boldsymbol{t}^{(i)}$, rainfall $\boldsymbol{r}^{(i)}$ and weather day type $\boldsymbol{d}^{(i)}$ as inputs. According to [38], different time steps and different features of the same time step have unequal effects on the output. In order to simultaneously enhance the sensitivity of features and the temporal dimension while making short-term load predictions, this paper proposes a novel attention mechanism named dual-stage attention mechanism (DA) that combines a feature attention (FA) and a temporal attention mechanism (TA). Combining the advantages of TA and FA, DA can fully capture the relationship between variables and temporal dependence in load time series, so as to provide data support for an efficient forecasting model.

3.2.1. Feature Attention Mechanism

To highlight the more critical influence features of the input, FA is introduced. As the successor of the attention mechanism, FA also has three parts, and the calculation of the attention weight of the input features is realized through the neural network in this paper. The realization of FA is demonstrated in Figure 3 and as below.

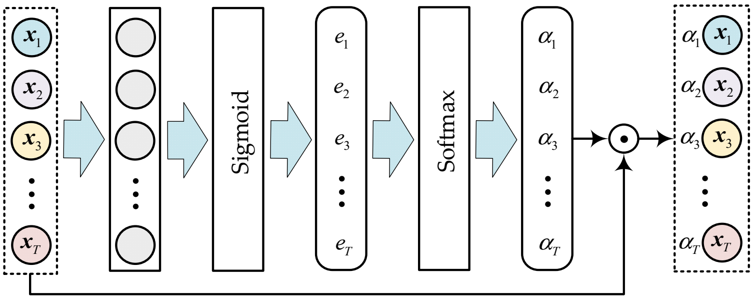

Figure 3. Feature attention mechanism.

(1) Attention weight calculation: Set $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T) \in \mathbb{R}^{N\times T}$ as the input vector of the prediction model, where $\boldsymbol{x}_j = \left\{x_j^{(i)}\right\}_{i=1}^{N} \in \mathbb{R}^N (j \in \left\{1, 2, \ldots, n_f\right\})$ and N is the number of samples. The quantization process of the corresponding weight of each feature is shown in Equation (10).

$$\boldsymbol{e} = \sigma(\boldsymbol{X}\boldsymbol{W}_e + \boldsymbol{b}_e) \tag{10}$$

Among them, $\boldsymbol{e} = (e_1, e_2, \ldots, e_T)$ is the unnormalized attention weight, $\boldsymbol{W}_e \in \mathbb{R}^{T\times T}$ is the trainable coefficient matrix and $\boldsymbol{b}_e \in \mathbb{R}^T$ is the bias.

(2) Weight normalization: In order to make the attention weight satisfy the probability distribution whose sum is 1, $\boldsymbol{e}$ is normalized by the softmax function, as shown in Equation (11).

$$\alpha_j = \mathrm{softmax}(e_j) = \exp(e_j) / \sum_{i=1}^{T} \exp(e_i) \tag{11}$$

(3) Intermediate semantic vector calculation: The normalized weight α_j is multiplied by the corresponding feature vector $\boldsymbol{x}_j$ to achieve the purpose of enhancing or reducing the expression of $\boldsymbol{x}_j$. Finally, the adaptively optimized feature vector $\boldsymbol{X}^{\mathrm{ATT}}$ can be obtained as shown in Equation (12).

$$\boldsymbol{X}^{\mathrm{ATT}} = (\alpha_1 \boldsymbol{x}_1, \alpha_2 \boldsymbol{x}_2, \ldots, \alpha_T \boldsymbol{x}_T) \tag{12}$$

It is worth noting that the attention weights are dynamically changed during the training process of the model, and the weights are determined only when the iterations converge.

3.2.2. Temporal Attention Mechanism

In order to capture the temporal correlation relationship between each time step in $\boldsymbol{X}^{\mathrm{ATT}}$ and the current prediction results, TA is introduced. The adaptive extraction of features at important moments is realized by integrating TA and the BiGRU network. The implementation is shown in Figure 4 and as below.

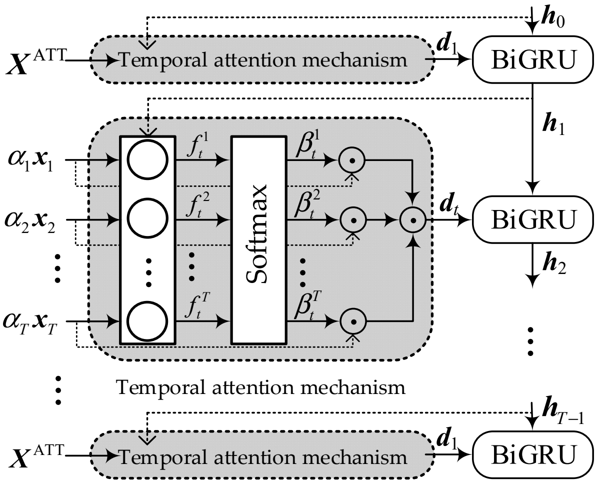
$\boldsymbol{h}_0$
$\boldsymbol{X}^{\mathrm{ATT}}$
Temporal attention mechanism
$\boldsymbol{d}_1$
BiGRU
$\boldsymbol{h}_1$
$\alpha_1\boldsymbol{x}_1$
f_t^1
β_t^1
$\alpha_2\boldsymbol{x}_2$
f_t^2
Softmax
β_t^2
$\boldsymbol{d}_t$
BiGRU
$\boldsymbol{h}_2$
$\alpha_T\boldsymbol{x}_T$
f_t^T
β_t^T
Temporal attention mechanism
$\boldsymbol{h}_{T-1}$
$\boldsymbol{X}^{\mathrm{ATT}}$
Temporal attention mechanism
$\boldsymbol{d}_1$
BiGRU

Figure 4. Temporal attention mechanism.

TA also has three parts, which are the same as FA.

(1) Attention weight calculation: Taking the vector $\boldsymbol{X}^{\mathrm{ATT}}$ containing the feature association relationship and the hidden state $\boldsymbol{h}_{t-1}$ at the previous time step of BiGRU as the input of TA, the attention weights at t-th time step in the iterative process are quantified, as shown in Equation (13).

$$\boldsymbol{f}_t = \sigma(\left[\boldsymbol{X}^{\mathrm{ATT}}; \boldsymbol{h}_{t-1}\right] \boldsymbol{W}_f + \boldsymbol{b}_f) \tag{13}$$

where $\boldsymbol{W}_f \in \mathbb{R}^{(2T) \times n_p}$ is the trainable coefficient matrix, $\boldsymbol{b}_f \in \mathbb{R}^{n_p}$ is the bias and n_p is the number of hidden elements in the last layer of BiGRU.

(2) Weight normalization: In addition, $\boldsymbol{f}_t$ is normalized using the softmax function, as shown in Equation (14).

$$\beta_t^j = \exp(f_t^j) / \sum_{i=1}^{T} \exp(f_t^i) \tag{14}$$

(3) Intermediate semantic vector calculation: In order to obtain the implicit temporal correlation relationship at t-th time step, β_t^j and $\alpha_t \boldsymbol{x}_t$ are weighted and summed to obtain an intermediate semantic vector $\boldsymbol{d}_t$, which contains features and temporal-related information, as shown in Equation (15).

$$\boldsymbol{d}_t = \sum_{j=1}^{T} \beta_t^j \alpha_j \boldsymbol{x}_j \tag{15}$$

Once the iteration of BiGRU terminates, the final hidden state $\boldsymbol{h}_T$ and the intermediate semantic vector $\boldsymbol{d}_T$ are obtained. The final prediction is taken by a single-layer feedforward network utilizing $\boldsymbol{h}_T$ and $\boldsymbol{d}_T$, as shown in Equation (16).

$$\boldsymbol{Y}' = [\boldsymbol{h}_T; \boldsymbol{d}_T]\boldsymbol{W}_y + \boldsymbol{b}_y \tag{16}$$

where $\boldsymbol{W}_y \in \mathbb{R}^{(2T)\times n_p}$ is the weight metrices of the feedforward network and $\boldsymbol{b}_y \in \mathbb{R}^{n_p}$ is the bias.

Assuming that all parameters of the DA-BiGRU model are $\boldsymbol{\theta}$, its loss function is shown in Equation (17). With the goal of minimizing this loss function, the final DA-BiGRU prediction model is obtained when the training is over.

$$J(\boldsymbol{Y}', \boldsymbol{Y}, \boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{Y}'^{(i)}, \boldsymbol{Y}^{(i)}\right)^2 \tag{17}$$

3.3. CS-GWO Optimization Algorithm

Facts have proved that the prediction model trained with an Adam optimizer can achieve rapid convergence in the early stage, but the learning rate is too low in the later stage of training, which may affect the effective convergence of the model and cause generalization problems [39]. Swarm intelligence optimization algorithms (e.g., PSO, GWO) are used to address the aforementioned problems; however, this leads to another problem of failing to find global optimal solutions [16]. In Ref. [40], an improved GWO algorithm created by incorporating a crisscross optimization algorithm (CSO) [41] is proposed for solving the optimal power flow problem, effectively avoiding falling into the local optimum and preventing the premature convergence. Inspired by this, this paper applies a crisscross grey wolf optimizer (CS-GWO) to optimize the DA-BiGRU model in the early stage of training, so as to further accelerate the convergence and improve the generation of the short-term load prediction model. Compared with GWO, CS-GWO achieves better global search ability and group diversity by introducing horizontal crossover and vertical crossover operators of CSO.

The implementation of CS-GWO is mainly composed of five parts, which are parameter initialization, hunting, attacking prey, horizontal crossover and vertical crossover. The implementation process is described in detail as follows.

3.3.1. Parameter Initialization

Set the population of grey wolf as $\boldsymbol{\Phi} = [\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_{n_M}]^T \in \mathbb{R}^{n_M \times D}$, where n_M is the population size and D is the population dimension. Select the individual with the best fitness value in population $\boldsymbol{\Phi}$ as grey wolf α, the individual with the second-best fitness value in population $\boldsymbol{\Phi}$ as grey wolf β, the individual with the third-best fitness value in population $\boldsymbol{\Phi}$ as grey wolf δ, and the rest of the population $\boldsymbol{\Phi}$ as grey wolf ω.

3.3.2. Hunting

Since the optimal hunting position of the wolves is unknown in the abstract search space, it is necessary to set three wolves with the strongest hunting ability to guide the hunting of the wolves. Assume that the three wolves are α, β and ω, respectively, and the other wolf ω updates its position according to the positions of the above three wolves during the iteration process, as shown in Equations (18) and (19).

$$\begin{cases} \boldsymbol{\theta}'_\alpha = \boldsymbol{\theta}_\alpha(t) - A_1 \cdot |C_1\boldsymbol{\theta}_\alpha(t) - \boldsymbol{\theta}_\omega(t)| \\ \boldsymbol{\theta}'_\beta = \boldsymbol{\theta}_\beta(t) - A_2 \cdot |C_2\boldsymbol{\theta}_\beta(t) - \boldsymbol{\theta}_\omega(t)| \\ \boldsymbol{\theta}'_\delta = \boldsymbol{\theta}_\delta(t) - A_3 \cdot |C_3\boldsymbol{\theta}_\delta(t) - \boldsymbol{\theta}_\omega(t)| \end{cases} \tag{18}$$

$$\boldsymbol{\theta}_\omega(t+1) = \frac{\boldsymbol{\theta}'_\alpha + \boldsymbol{\theta}'_\beta + \boldsymbol{\theta}'_\delta}{3} \tag{19}$$

where $\boldsymbol{\theta}_\alpha(t)$, $\boldsymbol{\theta}_\beta(t)$, $\boldsymbol{\theta}_\delta(t)$ and $\boldsymbol{\theta}_\omega(t)$ are the positions of grey wolves α, β, δ and ω at t-th iteration, respectively, and A_1, A_2, A_3, C_1, C_2 and C_3 are synergy coefficients, which are calculated as Equations (20) and (21).

3.3.3. Attack Prey

Once the prey is at rest, grey wolves stop searching and attack the prey. To simulate this process, GWO designs a synergy coefficient A, as shown in Equation (20).

$$A = 2a \cdot r_1 - a \tag{20}$$

where r_1 is a random number in the range of [0, 1], and a linearly decreases from 2 to 0 during the entire iteration process.

It can be seen from Equation (20) that GWO simulates the attack process of grey wolves. When $|A| < 1$, the wolves attack the prey; when $|A| > 1$, the wolves leave the prey alone, hoping to find better prey.

In addition, in order to avoid local optimum, GWO also designs another synergy coefficient C, as shown in Equation (21).

$$C = 2r_2 \tag{21}$$

where r_2 is a random number in the range of [0, 1].

3.3.4. Horizontal Crossover

In order to improve the global search ability of GWO, horizontal crossover (HC) is used to perform arithmetic crossover operations between two different individuals in all dimensions. Assuming that the i-th parent $\boldsymbol{\theta}_i$ and the j-th parent $\boldsymbol{\theta}_j(i,j \in \{1,2,\ldots,n_M\})$ perform HC operations on the d-th dimension, respectively, their offspring can be expressed as:

$$\begin{cases} \boldsymbol{\theta}_{i,d}^{HC} = r_3 \times \boldsymbol{\theta}_{i,d} + (1-r_3) \times \boldsymbol{\theta}_{j,d} + C_4 \times \left(\boldsymbol{\theta}_{i,d} - \boldsymbol{\theta}_{j,d}\right) \\ \boldsymbol{\theta}_{j,d}^{HC} = r_4 \times \boldsymbol{\theta}_{j,d} + (1-r_4) \times \boldsymbol{\theta}_{i,d} + C_5 \times \left(\boldsymbol{\theta}_{j,d} - \boldsymbol{\theta}_{i,d}\right) \end{cases} \tag{22}$$

where r_3 and r_4 are uniformly distributed random values in the range of [0, 1], and C_4 and C_5 are uniformly distributed random values in the range of [−1, 1]. Once the HC is over, the new population $\boldsymbol{\Phi}^{HC} = \left[\boldsymbol{\theta}_1^{HC}, \boldsymbol{\theta}_2^{HC}, \ldots, \boldsymbol{\theta}_{n_M}^{HC}\right]^T \in \mathbb{R}^{n_M \times D}$ can be obtained.

3.3.5. Vertical Crossover

In order to improve the population diversity of GWO, vertical crossover (VC) is used to perform arithmetic crossover operations for all individuals between two different dimensions to generate offspring. Assuming that the d_1-th dimension and d_2-th dimension of the individual perform VC operations, the offspring can be expressed as:

$$\boldsymbol{\theta}_{i,d_1}^{VC} = r \times \boldsymbol{\theta}_{i,d_1} + (1-r) \times \boldsymbol{\theta}_{i,d_2} \tag{23}$$

where r is a random value uniformly distributed in the range of [0, 1]. Once the VC is over, a new population $\boldsymbol{\Phi}^{VC} = \left[\boldsymbol{\theta}_1^{VC}, \boldsymbol{\theta}_2^{VC}, \ldots, \boldsymbol{\theta}_{n_M}^{VC}\right]^T \in \mathbb{R}^{n_M \times D}$ is obtained.

3.3.6. The Detailed Implementation Steps of CS-GWO

To solve the problems of gradient disappearance and gradient explosion problems in deep neural networks [42], the CS-GWO is used to optimize weights and bias $\boldsymbol{\theta}$ of the DA-GRU model in the early stage of training, aiming to improve the generalization performance of the model. The flow chart of CS-GWO is shown in Figure 5, and the detailed implementation steps of the CS-GWO algorithm are as follows:

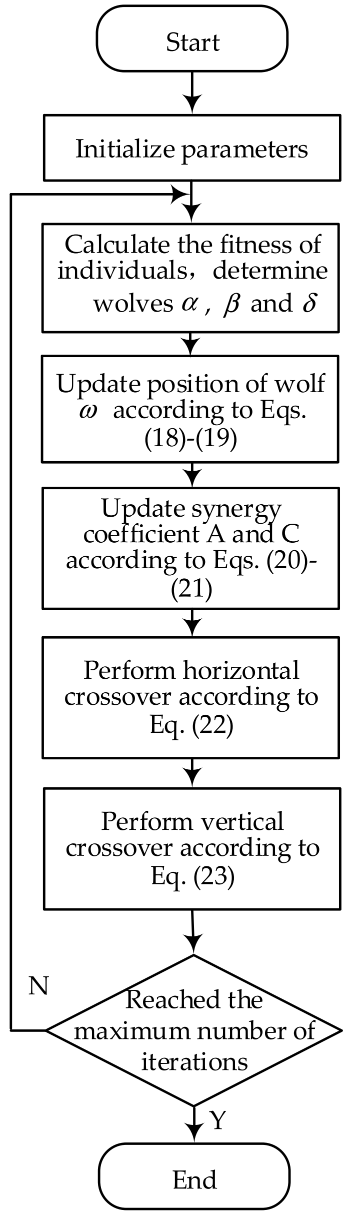
Start
Initialize parameters
Calculate the fitness of individuals，determine wolves α , β and δ
Update position of wolf ω according to Eqs. (18)-(19)
Update synergy coefficient A and C according to Eqs. (20)-(21)
Perform horizontal crossover according to Eq. (22)
Perform vertical crossover according to Eq. (23)
Reached the maximum number of iterations
N
Y
End

Figure 5. The flow chart of CS-GWO.

(1) Initialize parameters: Set the number of grey wolf populations n_M, the maximum number of iterations T and the population dimension D (the number of weights and bias of the DA-BiGRU model), and initialize the population $\mathbf{\Phi}$.

(2) Set the fitness function: Take Equation (17) as the fitness function.

(3) Determine wolves α, β **and** δ**:** Use Equation (17) to calculate the fitness of each individual; the individual with the best fitness value is grey wolf α, the second-best fitness value is grey wolf β and the third-best fitness value is grey wolf ω.

(4) Update position and synergy coefficient: Firstly, update the position of grey wolf ω according to Equations (18) and (19), and then update A and C according to Equations (20) and (21).

(5) Horizontal crossover: According to Equation (22), perform horizontal crossover on the parent population $\mathbf{\Phi}$ to obtain the offspring population $\mathbf{\Phi}^{HC}$, and use Equation (17) to calculate the fitness of each individual. If the fitness of individual $\boldsymbol{\theta}_k (k \in \{1, 2, \ldots, n_M\})$ in $\mathbf{\Phi}$ is worse than that of individual $\boldsymbol{\theta}_k^{HC}$ in $\mathbf{\Phi}^{HC}$, replace $\boldsymbol{\theta}_k$ with $\boldsymbol{\theta}_k^{HC}$; otherwise, do not replace.

(6) Vertical crossover: According to Equation (23), carry out horizontal crossover on the parent population $\boldsymbol{\Phi}$ to obtain the offspring population $\boldsymbol{\Phi}^{VC}$, and use Equation (17) to calculate the fitness of each individual. If the fitness of individual $\boldsymbol{\theta}_k$ in $\boldsymbol{\Phi}$ is worse than that of individual $\boldsymbol{\theta}_k^{VC}$ in $\boldsymbol{\Phi}^{VC}$, replace $\boldsymbol{\theta}_k$ with $\boldsymbol{\theta}_k^{VC}$; otherwise, do not replace.

(7) Iteration termination: If the number of iterations reaches T, the position of grey wolf α is used as the initial weight and threshold $\boldsymbol{\theta}_\alpha$ of the DA-GRU model; otherwise, return to step (3) and continue the iteration.

4. Evaluation Index

In order to evaluate the effectiveness of the proposed prediction model, this paper uses root mean square error (RMSE), mean absolute error (MAE), symmetric mean absolute percentage error (SMAPE) and decision coefficient (R^2) to evaluate the prediction results. The definitions of the four evaluation indicators are shown in Equations (24)–(27), where RMSE, MAE and SMAPE are indicators used to describe the error between the predicted value and the real value. The smaller the value, the more accurate the prediction result. Furthermore, R^2 is the indicator used to assess the linear relationship between the input and output values. The larger the value, the higher the prediction accuracy.

$$\text{RMSE} = \sqrt{\frac{1}{n_{test}} \sum_{n_{test}} \left(\boldsymbol{Y}_{test} - \hat{\boldsymbol{Y}}_{test}\right)^2} \tag{24}$$

$$\text{MAE} = \frac{1}{n_{test}} \sum_{n_{test}} \left|\boldsymbol{Y}_{test} - \hat{\boldsymbol{Y}}_{test}\right| \tag{25}$$

$$\text{SMAPE} = \frac{1}{n_{test}} \sum_{n_{test}} \frac{\left|\boldsymbol{Y}_{test} - \hat{\boldsymbol{Y}}_{test}\right|}{\left(\boldsymbol{Y}_{test} + \hat{\boldsymbol{Y}}_{test}\right)/2} \tag{26}$$

$$\text{R}^2 = 1 - \frac{\sum\limits_{n_{test}} \left(\boldsymbol{Y}_{test} + \hat{\boldsymbol{Y}}_{test}\right)^2}{\sum\limits_{n_{test}} \left(\boldsymbol{Y}_{test} - \overline{\boldsymbol{Y}}_{test}\right)^2} \tag{27}$$

where $\boldsymbol{Y}_{test}$ and $\hat{\boldsymbol{Y}}_{test}$ are the actual and predicted load values in the testing dataset, respectively, $\overline{\boldsymbol{Y}}_{test}$ is the average value of the actual load value and n_{test} is the sample number of the testing dataset.

5. Experiment and Analysis

This section will verify the effectiveness of the proposed DA-CS-GWO-BiGRU short-term load forecasting model through four evaluation indicators (RMSE, MAE, SMAPE and R^2) and two experiments. In addition, in order to reduce the errors caused by the experimental operation, both experiments were performed 20 times, and the average value was taken as the final experimental result. Both experiments are based on Python 3.8 and the Keras deep learning library. The core configuration of the used computer is Intel (R) Core (TM) i5-9600K 6-core processor, 3.70 GHz operating frequency, 8 GB memory capacity and Windows 10 operating system.

Particularly, the load data used in this paper is the real sample data of a region in 2018. These sample data have a total of 365 pieces, and their time resolution is 24. In order to reduce the influence of data distribution on the experimental results, the data is randomly sorted; 300 samples are selected as the training dataset, 30 samples are used as the validation dataset and 35 samples are used as the testing dataset. These datasets are depicted in Figure 6.

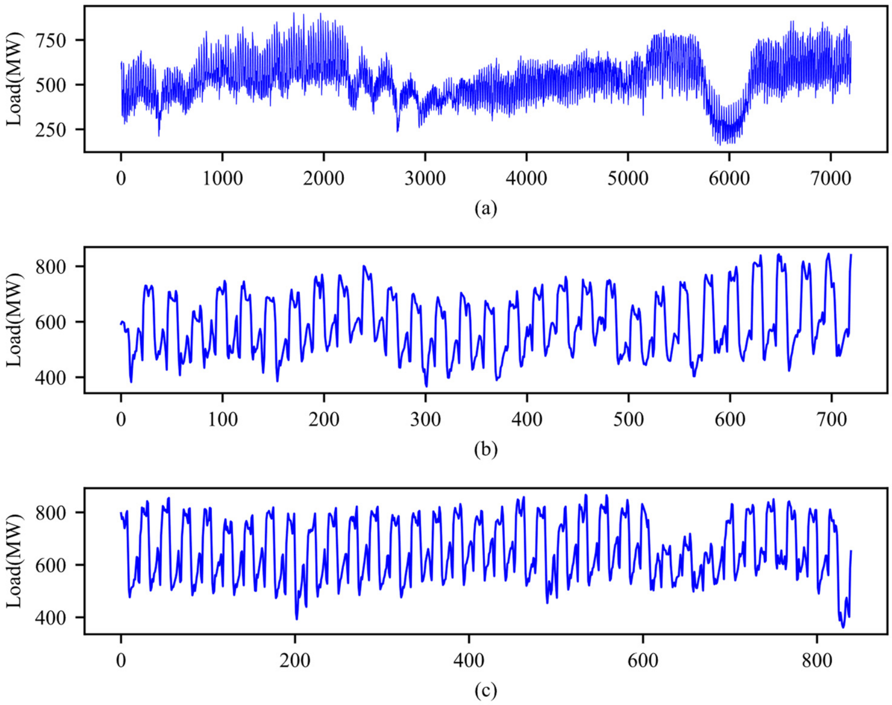

Figure 6. Training, validation and testing datasets used in this paper: (**a**) training dataset, (**b**) validation dataset, (**c**) testing dataset.

5.1. Parameter Settings

The attention mechanism is realized by a single-layer fully connected neural network, the number of neurons is 32 and the activation function is softmax.

The number of neurons in the hidden layer of the prediction model based on a BP neural network is 32, the activation function is ReLU, the number of neurons in the output layer is 24 and the activation function is linear. The unit number of the models based on GRU and BiGRU is 32, the number of neurons in the output layer is 24 and the activation function is linear. The BP, GRU, BiGRU, FA-BiGRU, TA-BiGRU, and DA-BiGRU models all use the Adam optimizer, and their hyperparameters β_1, β_2 and ε are set to 0.9, 0.999 and 1×10^{-8}, respectively. In addition, MSE is used as the loss function, and the number of iterations of these models is set to 500.

5.2. Case 1: The Effectiveness of the BiGRU Model and Dual-Stage Mechanism

In order to better evaluate the effectiveness of the DA-BiGRU prediction model proposed in this paper, this section verifies the superiority of the BiGRU model and effectiveness of the dual-stage attention mechanism from the aspect of short-term load prediction. Persistence, BP, GRU, BiGRU, feature-attention-mechanism-based BiGRU (FA-BiGRU) and temporal-attention-mechanism-based BiGRU (TA-BiGRU) models are compared with the DA-BiGRU model in this case. The experimental results are shown in Table 1 and Figure 7, where Figure 7 is a comparison chart between the prediction results of different models and the real values on 29–31 December 2018.

As shown in Table 1 and Figure 7, the following conclusions can be drawn:

(1) The advantages of BiGRU model:

The prediction performance of the deep learning model is the best among the single prediction models (i.e., persistence, BP, GRU and BiGRU models), and the prediction accuracy of the BiGRU model is the highest. For example, compared with the classic baseline model persistence, the RMSE, MAE and SMAPE values of the BiGRU model are reduced by 15.46%, 14.38% and 0.942%, respectively, and the R^2 value is increased by 1.67%. Compared with the shallow neural network BP model, the RMSE, MAE and SMAPE values of the BiGRU model are reduced by 10.56%, 9.40% and 0.531%, respectively, and the R^2

value is increased by 1.98%. In addition, compared with the GRU model, the BiGRU model has the best RMSE, MAE, SMAPE and R^2 values.

Table 1. The experiment results of case study 1.

Prediction Model	RMSE/MW	MAE/MW	SMAPE	R^2
persistence	36.679	29.258	4.810	0.900
BP	34.671	27.650	4.399	0.911
GRU	32.141	25.128	3.892	0.924
BiGRU	31.009	25.051	3.868	0.929
FA-BiGRU	29.583	23.068	3.591	0.935
TA-BiGRU	29.840	24.239	3.830	0.930
DA-BiGRU	29.053	22.897	3.566	0.937

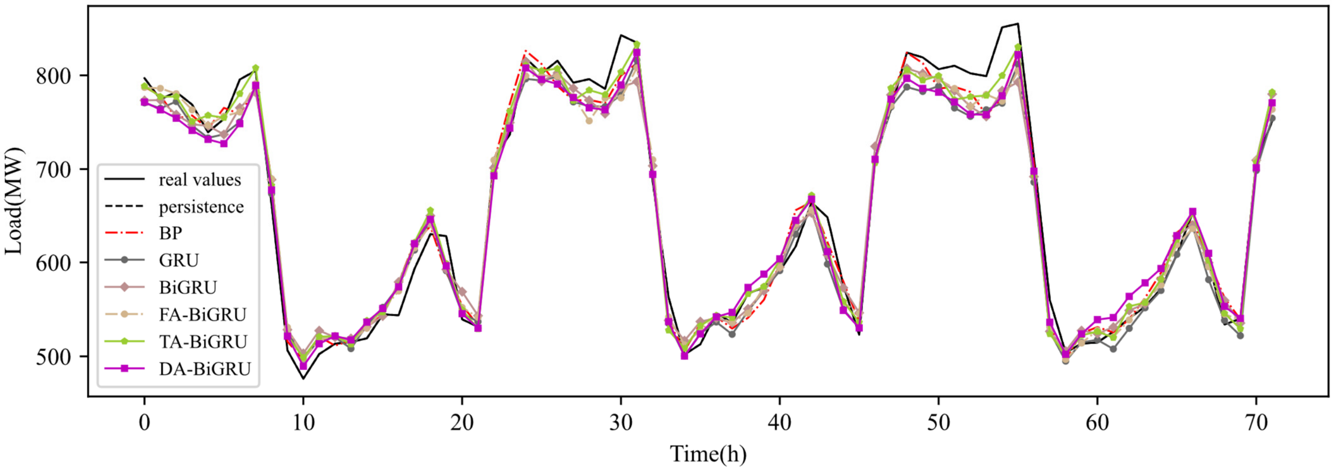

Figure 7. Case study 1: comparison of forecasting results on 29–31 December 2018.

The reasons are as follows: Firstly, the machine learning model uses a large amount of historical load data for training, which can effectively capture the nonlinear relationship of load time series, and the prediction performance is improved compared with the persistence model. Secondly, the BiGRU model has a unique bidirectional propagation structure, which can link the past and future influencing factors with the current load time series so as to improve the accuracy of short-term load forecasting.

(2) The effectiveness of the dual-stage attention mechanism:

This model combined with the feature attention mechanism can automatically extract the correlation between each feature, which has the ability of reducing the prediction error of the model. Compared with the BiGRU model, the RMSE, MAE and SMAPE values of the FA-BiGRU model decreased by 4.60%, 7.92% and 7.16%, respectively, and the R^2 value increased by 0.65%.

This model combined with the temporal attention mechanism realizes the adaptive extraction of features at important moments, which improves the prediction stability of the model. Compared with the BiGRU model, the RMSE, MAE and SMAPE values of the TA-BiGRU model decreased by 3.77%, 3.24% and 0.98%, respectively, and the R^2 value increased by 0.11%.

In addition, compared with other comparison models in this case study, the proposed DA-BiGRU model has the best RMSE, MAE, MAPE and R^2 values. This is because this model combines feature and temporal attention mechanisms, which can improve the sensitivity of the model to key features and key time steps, and finally achieve the purpose of improving prediction accuracy.

5.3. Case 2: The Effectiveness of the CS-GWO Algorithm

The popular suite of benchmark functions in validating optimization performance, i.e., CEC 2017 [43], is utilized in this subsection to conduct extensive optimization experiments. The CEC 2017 test suite has 30 functions, which can be divided into four categories: unimodal functions (F_1–F_3), multimodal functions (F_4–F_{10}), hybrid functions (F_{11}–F_{20}) and composition functions (F_{21}–F_{30}). The ideal optimal value of each benchmark functions is 0.

Moreover, the well-known optimization algorithms (i.e., PSO, WOA, GWO and CSO) are compared with CS-GWO to evaluate the effectiveness of the CS-GWO algorithm from various perspectives, including accuracy, the Wilcoxon signed-rank test and a paired samples *t*-test.

5.3.1. The Setting of the Numerical Experiments

The dimension of benchmark functions is uniformly set to 30 in this subsection. For a fair comparison, the number of iterations of the swarm intelligence optimization algorithm and the number of individuals is set to 3000 and 30, respectively. The position of PSO is set in the range of $[-1, 1]$, and the limit of speed is set in the range of $[-0.5, 0.5]$ [44]. For WOA [45], GWO [46] and CS-GWO, the limit of individual is set in the range of $[-1, 1]$. Among them, the vertical crossover probability of CSO and CS-GWO is set to 60%, and the horizontal intersection crossover is set to 100% [47]. To reduce statistical errors, all the reported results in this subsection are based on 30 independent runs.

5.3.2. The Comparison of Optimization Accuracy

The above-mentioned algorithms are evaluated using the CEC 2017 test suite, and the experimental results are shown in Table 2. The reported values in Table 2 are based on the errors between the terminated values of the optimization process and the target values of the benchmark functions. To intuitively quantify the optimization ability of the metaheuristics, mean values (Mean), minimum values (Min), maximum values (Max), standard deviation (Std) and ranks (Rank) are used to evaluate the accuracy. Mean, Min and Max reveal the optimization accuracy of the algorithm, Std reveals the optimization stability and Rank is based on the Friedman test [48] to rank the optimization performance of the algorithm from the aspect of statistics. Moreover, the minimum values of Mean in each benchmark function are shown in bold.

Table 2. Comparison of CS-GWO with well-known algorithms for CEC 2017 test functions.

Functions	Metrics	PSO	WOA	GWO	CSO	CS-GWO
F_1	Mean	3.169×10^{11}	$\mathbf{2.408 \times 10^{3}}$	3.781×10^{10}	2.275×10^{10}	4.875×10^{3}
	Min	1.137×10^{11}	1.070×10^{2}	5.121×10^{9}	4.985×10^{9}	1.000×10^{2}
	Max	5.806×10^{11}	9.847×10^{3}	8.496×10^{10}	6.467×10^{10}	1.771×10^{4}
	Std	9.145×10^{10}	2.219×10^{3}	2.197×10^{10}	1.430×10^{10}	5.283×10^{3}
	Rank	5	1	4	3	2
F_2	Mean	8.880×10^{2}	7.076×10^{2}	6.260×10^{2}	6.012×10^{2}	$\mathbf{5.550 \times 10^{2}}$
	Min	8.253×10^{2}	6.184×10^{2}	5.892×10^{2}	5.705×10^{2}	5.129×10^{2}
	Max	9.519×10^{2}	8.124×10^{2}	7.649×10^{2}	6.601×10^{2}	6.350×10^{2}
	Std	3.825×10^{1}	4.662×10^{1}	3.205×10^{1}	2.046×10^{1}	3.437×10^{1}
	Rank	5	4	3	2	1
F_3	Mean	1.406×10^{5}	5.016×10^{4}	4.718×10^{4}	3.481×10^{4}	$\mathbf{3.001 \times 10^{2}}$
	Min	8.673×10^{4}	2.929×10^{4}	3.286×10^{4}	1.132×10^{4}	3.000×10^{2}
	Max	3.292×10^{5}	7.088×10^{4}	6.894×10^{4}	6.353×10^{4}	3.003×10^{2}
	Std	6.473×10^{4}	1.017×10^{4}	9.881×10^{3}	1.028×10^{4}	6.049×10^{-2}
	Rank	5	4	3	2	1

Table 2. *Cont.*

Functions	Metrics	PSO	WOA	GWO	CSO	CS-GWO
F_4	Mean	6.990×10^{3}	**4.687×10^{2}**	6.504×10^{2}	5.645×10^{2}	4.937×10^{2}
	Min	2.951×10^{3}	4.001×10^{2}	5.166×10^{2}	4.833×10^{2}	4.641×10^{2}
	Max	1.422×10^{4}	4.911×10^{2}	1.218×10^{3}	7.337×10^{2}	5.187×10^{2}
	Std	2.613×10^{3}	1.995×10^{1}	1.457×10^{2}	6.179×10^{1}	1.520×10^{1}
	Rank	5	1	4	3	2
F_5	Mean	8.966×10^{2}	7.087×10^{2}	6.214×10^{2}	5.966×10^{2}	**5.774×10^{2}**
	Min	8.469×10^{2}	6.323×10^{2}	5.671×10^{2}	5.681×10^{2}	5.202×10^{2}
	Max	9.739×10^{2}	7.816×10^{2}	6.944×10^{2}	6.708×10^{2}	6.720×10^{2}
	Std	3.825×10^{1}	4.073×10^{1}	2.883×10^{1}	2.248×10^{1}	4.910×10^{1}
	Rank	5	4	3	2	1
F_6	Mean	7.009×10^{2}	6.730×10^{2}	6.270×10^{2}	6.196×10^{2}	**6.003×10^{2}**
	Min	6.795×10^{2}	6.482×10^{2}	6.126×10^{2}	6.102×10^{2}	6.000×10^{2}
	Max	7.331×10^{2}	7.250×10^{2}	6.460×10^{2}	6.366×10^{2}	6.017×10^{2}
	Std	1.221×10^{1}	1.364×10^{1}	8.497×10^{0}	6.550×10^{0}	4.418×10^{-1}
	Rank	5	4	3	2	1
F_7	Mean	1.397×10^{3}	1.140×10^{3}	8.841×10^{2}	**8.559×10^{2}**	8.710×10^{2}
	Min	1.218×10^{3}	1.022×10^{3}	8.019×10^{2}	7.960×10^{2}	7.698×10^{2}
	Max	1.525×10^{3}	1.324×10^{3}	1.076×10^{3}	9.970×10^{2}	8.977×10^{2}
	Std	6.925×10^{1}	7.795×10^{1}	6.307×10^{1}	5.190×10^{1}	3.069×10^{1}
	Rank	5	4	3	1	2
F_8	Mean	1.103×10^{3}	9.403×10^{2}	9.039×10^{2}	8.979×10^{2}	**8.742×10^{2}**
	Min	1.002×10^{3}	9.114×10^{2}	8.541×10^{2}	8.479×10^{2}	8.129×10^{2}
	Max	1.164×10^{3}	9.910×10^{2}	9.464×10^{2}	1.032×10^{3}	9.867×10^{2}
	Std	3.661×10^{1}	2.327×10^{1}	2.433×10^{1}	3.537×10^{1}	4.819×10^{1}
	Rank	5	4	3	2	1
F_9	Mean	9.729×10^{3}	7.439×10^{3}	2.287×10^{3}	2.092×10^{3}	**9.002×10^{2}**
	Min	5.733×10^{3}	3.310×10^{3}	1.410×10^{3}	1.035×10^{3}	9.000×10^{2}
	Max	1.324×10^{4}	1.376×10^{4}	4.608×10^{3}	3.691×10^{3}	9.029×10^{2}
	Std	1.659×10^{3}	3.347×10^{3}	7.803×10^{2}	7.556×10^{2}	5.347×10^{-1}
	Rank	5	4	3	2	1
F_{10}	Mean	8.429×10^{3}	6.333×10^{3}	5.139×10^{3}	**4.483×10^{3}**	7.671×10^{3}
	Min	6.359×10^{3}	4.002×10^{3}	2.849×10^{3}	3.091×10^{3}	6.807×10^{3}
	Max	1.005×10^{4}	9.761×10^{3}	8.679×10^{3}	7.903×10^{3}	8.508×10^{3}
	Std	8.850×10^{2}	1.699×10^{3}	1.674×10^{3}	1.270×10^{3}	4.446×10^{2}
	Rank	5	3	2	1	4
F_{11}	Mean	8.717×10^{3}	1.220×10^{3}	2.183×10^{3}	1.786×10^{3}	**1.159×10^{3}**
	Min	4.894×10^{3}	1.153×10^{3}	1.389×10^{3}	1.276×10^{3}	1.108×10^{3}
	Max	1.418×10^{4}	1.298×10^{3}	4.783×10^{3}	3.937×10^{3}	1.217×10^{3}
	Std	2.371×10^{3}	3.880×10^{1}	9.575×10^{2}	7.822×10^{2}	3.555×10^{1}
	Rank	5	2	4	3	1
F_{12}	Mean	2.599×10^{10}	**1.764×10^{5}**	3.831×10^{8}	5.773×10^{8}	3.160×10^{5}
	Min	2.572×10^{9}	1.191×10^{4}	2.075×10^{7}	2.392×10^{7}	2.261×10^{4}
	Max	2.031×10^{11}	8.030×10^{5}	1.507×10^{9}	3.799×10^{9}	1.485×10^{6}
	Std	3.719×10^{10}	1.616×10^{5}	3.689×10^{8}	8.210×10^{8}	3.465×10^{5}
	Rank	5	1	4	3	2
F_{13}	Mean	1.224×10^{10}	1.733×10^{4}	1.497×10^{8}	4.750×10^{7}	**1.410×10^{4}**
	Min	2.027×10^{8}	3.680×10^{3}	3.990×10^{4}	4.008×10^{4}	1.416×10^{3}
	Max	2.020×10^{11}	4.611×10^{4}	1.498×10^{9}	1.405×10^{9}	4.460×10^{4}
	Std	3.616×10^{10}	1.109×10^{4}	4.094×10^{8}	2.564×10^{8}	1.138×10^{4}
	Rank	5	2	4	3	1

Table 2. *Cont.*

Functions	Metrics	PSO	WOA	GWO	CSO	CS-GWO
F_{14}	Mean	4.157×10^{6}	**1.219×10^{4}**	4.015×10^{5}	1.794×10^{5}	4.376×10^{4}
	Min	1.956×10^{4}	1.746×10^{3}	2.666×10^{4}	2.325×10^{3}	4.285×10^{3}
	Max	4.485×10^{7}	1.370×10^{5}	1.337×10^{6}	9.272×10^{5}	2.532×10^{5}
	Std	8.338×10^{6}	2.417×10^{4}	4.196×10^{5}	2.868×10^{5}	5.106×10^{4}
	Rank	5	1	4	3	2
F_{15}	Mean	1.424×10^{9}	8.116×10^{3}	4.410×10^{6}	4.594×10^{6}	**3.994×10^{3}**
	Min	2.520×10^{5}	1.731×10^{3}	2.667×10^{4}	1.355×10^{4}	1.607×10^{3}
	Max	1.273×10^{10}	3.219×10^{4}	3.595×10^{7}	9.453×10^{7}	2.100×10^{4}
	Std	2.717×10^{9}	8.011×10^{3}	9.902×10^{6}	1.740×10^{7}	4.244×10^{3}
	Rank	5	2	4	3	1
F_{16}	Mean	5.003×10^{3}	2.990×10^{3}	2.559×10^{3}	**2.455×10^{3}**	2.489×10^{3}
	Min	3.196×10^{3}	2.365×10^{3}	2.069×10^{3}	2.033×10^{3}	1.700×10^{3}
	Max	1.215×10^{4}	3.871×10^{3}	3.254×10^{3}	3.319×10^{3}	3.017×10^{3}
	Std	2.024×10^{3}	3.759×10^{2}	3.002×10^{2}	3.224×10^{2}	3.925×10^{2}
	Rank	5	4	3	1	2
F_{17}	Mean	5.052×10^{3}	3.017×10^{3}	2.460×10^{3}	2.392×10^{3}	**2.126×10^{3}**
	Min	3.917×10^{3}	2.269×10^{3}	2.102×10^{3}	1.990×10^{3}	1.612×10^{3}
	Max	1.022×10^{4}	3.507×10^{3}	3.213×10^{3}	3.419×10^{3}	3.014×10^{3}
	Std	1.165×10^{3}	3.432×10^{2}	2.835×10^{2}	3.141×10^{2}	3.478×10^{2}
	Rank	5	4	3	2	1
F_{18}	Mean	1.149×10^{8}	7.835×10^{5}	1.858×10^{6}	1.463×10^{6}	**1.632×10^{5}**
	Min	2.326×10^{6}	8.942×10^{4}	5.083×10^{4}	8.757×10^{4}	4.111×10^{4}
	Max	7.292×10^{8}	2.771×10^{6}	2.160×10^{7}	8.673×10^{6}	4.130×10^{5}
	Std	2.234×10^{8}	6.988×10^{5}	4.064×10^{6}	1.773×10^{6}	8.817×10^{4}
	Rank	5	4	3	2	1
F_{19}	Mean	1.516×10^{9}	9.880×10^{3}	1.345×10^{7}	2.830×10^{6}	**7.321×10^{3}**
	Min	8.062×10^{6}	1.979×10^{3}	3.550×10^{4}	6.985×10^{3}	1.991×10^{3}
	Max	2.216×10^{10}	4.385×10^{4}	3.380×10^{8}	1.373×10^{7}	3.052×10^{4}
	Std	4.402×10^{9}	9.797×10^{3}	6.135×10^{7}	3.282×10^{6}	7.050×10^{3}
	Rank	5	2	4	3	1
F_{20}	Mean	9.231×10^{3}	6.463×10^{3}	6.165×10^{3}	5.747×10^{3}	**4.678×10^{3}**
	Min	5.555×10^{3}	2.300×10^{3}	4.389×10^{3}	2.475×10^{3}	2.300×10^{3}
	Max	1.129×10^{4}	9.968×10^{3}	1.015×10^{4}	9.513×10^{3}	9.283×10^{3}
	Std	1.071×10^{3}	1.958×10^{3}	1.440×10^{3}	1.854×10^{3}	3.017×10^{3}
	Rank	5	4	3	2	1
F_{21}	Mean	2.697×10^{3}	2.520×10^{3}	2.408×10^{3}	**2.388×10^{3}**	2.403×10^{3}
	Min	2.583×10^{3}	2.411×10^{3}	2.373×10^{3}	2.341×10^{3}	2.319×10^{3}
	Max	2.838×10^{3}	2.650×10^{3}	2.523×10^{3}	2.436×10^{3}	2.469×10^{3}
	Std	5.997×10^{1}	6.193×10^{1}	2.773×10^{1}	2.040×10^{1}	3.968×10^{1}
	Rank	5	4	2	1	3
F_{22}	Mean	9.247×10^{3}	6.023×10^{3}	6.002×10^{3}	6.264×10^{3}	**3.598×10^{3}**
	Min	6.356×10^{3}	2.300×10^{3}	2.731×10^{3}	2.672×10^{3}	2.300×10^{3}
	Max	1.083×10^{4}	1.204×10^{4}	1.041×10^{4}	9.926×10^{3}	9.689×10^{3}
	Std	1.019×10^{3}	2.548×10^{3}	1.493×10^{3}	2.164×10^{3}	2.650×10^{3}
	Rank	5	4	2	3	1
F_{23}	Mean	3.363×10^{3}	3.479×10^{3}	2.792×10^{3}	2.757×10^{3}	**2.708×10^{3}**
	Min	3.098×10^{3}	3.095×10^{3}	2.726×10^{3}	2.701×10^{3}	2.674×10^{3}
	Max	3.869×10^{3}	3.882×10^{3}	2.946×10^{3}	2.895×10^{3}	2.767×10^{3}
	Std	1.838×10^{2}	1.772×10^{2}	5.363×10^{1}	3.990×10^{1}	2.849×10^{1}
	Rank	4	5	3	2	1

Table 2. *Cont.*

Functions	Metrics	PSO	WOA	GWO	CSO	CS-GWO
F_{24}	Mean	3.533×10^{3}	3.531×10^{3}	2.993×10^{3}	**2.947×10^{3}**	2.970×10^{3}
	Min	3.281×10^{3}	3.233×10^{3}	2.899×10^{3}	2.881×10^{3}	2.865×10^{3}
	Max	3.856×10^{3}	3.882×10^{3}	3.095×10^{3}	3.092×10^{3}	3.011×10^{3}
	Std	1.543×10^{2}	1.353×10^{2}	5.745×10^{1}	5.943×10^{1}	3.716×10^{1}
	Rank	4	5	3	1	2
F_{25}	Mean	4.003×10^{3}	2.915×10^{3}	3.005×10^{3}	2.960×10^{3}	**2.888×10^{3}**
	Min	3.581×10^{3}	2.884×10^{3}	2.930×10^{3}	2.906×10^{3}	2.883×10^{3}
	Max	5.042×10^{3}	2.948×10^{3}	3.220×10^{3}	3.044×10^{3}	2.910×10^{3}
	Std	3.285×10^{2}	2.514×10^{1}	7.924×10^{1}	3.339×10^{1}	4.344×10^{0}
	Rank	5	2	4	3	1
F_{26}	Mean	1.005×10^{4}	6.454×10^{3}	4.605×10^{3}	4.554×10^{3}	**4.105×10^{3}**
	Min	7.873×10^{3}	2.800×10^{3}	4.114×10^{3}	4.051×10^{3}	3.698×10^{3}
	Max	1.259×10^{4}	1.021×10^{4}	5.272×10^{3}	5.723×10^{3}	4.894×10^{3}
	Std	1.211×10^{3}	2.659×10^{3}	3.458×10^{2}	3.358×10^{2}	2.572×10^{2}
	Rank	5	4	2	3	1
F_{27}	Mean	3.833×10^{3}	4.218×10^{3}	3.200×10^{3}	**3.200×10^{3}**	3.211×10^{3}
	Min	3.410×10^{3}	3.644×10^{3}	3.200×10^{3}	3.200×10^{3}	3.201×10^{3}
	Max	5.619×10^{3}	4.903×10^{3}	3.200×10^{3}	3.200×10^{3}	3.221×10^{3}
	Std	4.177×10^{2}	3.348×10^{2}	2.205×10^{-4}	3.106×10^{-4}	5.428×10^{0}
	Rank	4	5	2	1	3
F_{28}	Mean	5.399×10^{3}	**3.157×10^{3}**	3.315×10^{3}	3.317×10^{3}	3.210×10^{3}
	Min	4.195×10^{3}	3.100×10^{3}	3.296×10^{3}	3.296×10^{3}	3.100×10^{3}
	Max	6.761×10^{3}	3.265×10^{3}	3.474×10^{3}	3.465×10^{3}	3.267×10^{3}
	Std	6.843×10^{2}	6.460×10^{1}	4.611×10^{1}	4.549×10^{1}	3.247×10^{1}
	Rank	5	1	3	4	2
F_{29}	Mean	3.507×10^{3}	3.498×10^{3}	2.987×10^{3}	2.963×10^{3}	**2.953×10^{3}**
	Min	3.197×10^{3}	3.312×10^{3}	2.878×10^{3}	2.877×10^{3}	2.860×10^{3}
	Max	3.771×10^{3}	3.695×10^{3}	3.111×10^{3}	3.074×10^{3}	2.992×10^{3}
	Std	1.371×10^{2}	9.597×10^{1}	6.437×10^{1}	6.320×10^{1}	3.351×10^{1}
	Rank	4	5	3	2	1
F_{30}	Mean	4.622×10^{9}	1.995×10^{4}	1.353×10^{7}	3.037×10^{7}	**8.942×10^{3}**
	Min	6.778×10^{7}	7.843×10^{3}	3.555×10^{4}	1.607×10^{4}	5.375×10^{3}
	Max	7.022×10^{10}	4.219×10^{4}	2.789×10^{8}	3.440×10^{8}	1.593×10^{4}
	Std	1.505×10^{10}	7.370×10^{3}	5.074×10^{7}	7.612×10^{7}	3.023×10^{3}
	Rank	5	2	3	4	1
Mean rank		4.883	3.183	3.117	2.316	1.500
Final rank		5	4	3	2	1

For the unimodal and functions (i.e., F_{1}–F_{10}), CS-GWO achieves the best results six times, and WOA and CSO share the remaining four best results. This reveals that WOA performs well for simple low-dimensional optimization problems. For the 10 hybrid functions (i.e., F_{11}–F_{20}), CS-GWO achieves the best performance seven times. Although WOA has the best values for F_{12} and F_{14}, it gets the second-worst values in 4 out of 10 cases. This reveals that WOA has unstable performance in solving complex problems [49,50]. For the 10 composition functions (i.e., F_{21}–F_{30}), CS-GWO achieves the best performance six times, followed by CSO's three times. The worst rank of CSO in composition functions is three with F_{21} and F_{27}, indicating that CSO has the ability to escape local optimum when applied in complex optimization problems [47].

Particularly, GWO never dominates in optimization of all benchmark functions, but it ranks third overall. This reveals that GWO has stable performance in solving optimization problems but easily falls into the optima trap [51]. Comprehensively speaking, CS-GWO ranks first overall among the optimization algorithms and obtains the best performance in 19 out of 30 functions, whether from the aspect of the optimization accuracy or stability.

From the above analysis, we can conclude that CS-GWO performs the best with the 30 dimensional optimization problems among all the compared algorithms. This is because CS-GWO combines the stable optimization performance of GWO and outstanding ability in finding global optima in the problem-solving space of CSO.

5.3.3. Wilcoxon Signed-Rank Test and Paired Samples *t*-Test

In order to further prove the validity of the CS-GWO algorithm, the parametric and non-parametric tests, which are the paired samples *t*-test (PSTT) [52] and Wilcoxon signed-rank test (WSRT) [53], are adopted to evaluate the difference of optimization performance between CS-GWO and the comparison algorithms in 30 benchmark functions. The null hypothesis of PSTT and WSRT are that there is no difference between two compared samples. If the difference approximately obeyed the normal distribution, PSTT is used. When this premise is not satisfied, WSRT can be selected. The results of PSTT and WSRT are shown in Tables 3 and 4, respectively.

Table 3. The paired samples *t*-test results.

Functions	CS-GWO–PSO		CS-GWO–WOA		CS-GWO–GWO		CS-GWO–CSO	
	t-Value	Sig. (2-Tailed)	*t*-Value	Sig. (2-Tailed)	*t*-Value	Sig. (2-Tailed)	*t*-Value	Sig. (2-Tailed)
F_1	-1.898×10^{1}	6.748×10^{-18}	2.413×10^{0}	2.236×10^{-2}	-8.712×10^{0}	1.367×10^{-9}	-9.427×10^{0}	$2.472 \times 10-1^{0}$
F_2	-3.355×10^{1}	9.349×10^{-25}	-1.603×10^{1}	6.012×10^{-16}	-7.118×10^{0}	7.828×10^{-8}	-7.771×10^{0}	1.434×10^{-8}
F_3	-3.733×10^{1}	4.521×10^{-26}	-1.447×10^{1}	8.498×10^{-15}	-2.535×10^{0}	1.688×10^{-2}	-4.732×10^{0}	5.343×10^{-5}
F_4	-1.361×10^{1}	4.016×10^{-14}	5.103×10^{0}	1.907×10^{-5}	-6.128×10^{0}	1.122×10^{-6}	-5.836×10^{0}	2.499×10^{-6}
F_5	-2.913×10^{1}	5.005×10^{-23}	-1.207×10^{1}	7.880×10^{-13}	-2.016×10^{0}	5.320×10^{-2}	-4.119×10^{0}	2.892×10^{-4}
F_6	-4.482×10^{1}	2.469×10^{-28}	-2.925×10^{1}	4.471×10^{-23}	-1.638×10^{1}	3.381×10^{-16}	-1.711×10^{1}	1.078×10^{-16}
F_7	-4.000×10^{1}	6.363×10^{-27}	-1.717×10^{1}	9.852×10^{-17}	-1.211×10^{0}	2.358×10^{-1}	-1.954×10^{0}	6.035×10^{-2}
F_8	-1.977×10^{1}	2.240×10^{-18}	-6.441×10^{0}	4.791×10^{-7}	-2.143×10^{0}	4.061×10^{-2}	-3.119×10^{0}	4.073×10^{-3}
F_9	-2.915×10^{1}	4.917×10^{-23}	-1.070×10^{1}	1.390×10^{-11}	-8.638×10^{0}	1.639×10^{-9}	-9.730×10^{0}	1.224×10^{-10}
F_{10}	-4.268×10^{0}	1.926×10^{-4}	3.982×10^{0}	4.199×10^{-4}	1.263×10^{1}	2.564×10^{-13}	7.920×10^{0}	9.815×10^{-9}
F_{11}	-1.744×10^{1}	6.491×10^{-17}	-6.545×10^{0}	3.613×10^{-7}	-4.355×10^{0}	1.517×10^{-4}	-5.824×10^{0}	2.584×10^{-6}
F_{12}	-3.827×10^{0}	6.381×10^{-4}	2.057×10^{0}	4.877×10^{-2}	-3.849×10^{0}	6.013×10^{-4}	-5.683×10^{0}	3.811×10^{-6}
F_{13}	-1.855×10^{0}	7.384×10^{-2}	-1.045×10^{0}	3.045×10^{-1}	-1.014×10^{0}	3.188×10^{-1}	-2.002×10^{0}	5.470×10^{-2}
F_{14}	-2.698×10^{0}	1.150×10^{-2}	2.856×10^{0}	7.859×10^{-3}	-2.508×10^{0}	1.800×10^{-2}	-4.504×10^{0}	1.004×10^{-4}
F_{15}	-2.870×10^{0}	7.579×10^{-3}	-2.481×10^{0}	1.915×10^{-2}	-1.445×10^{0}	1.592×10^{-1}	-2.437×10^{0}	2.118×10^{-2}
F_{16}	-6.619×10^{0}	2.964×10^{-7}	-5.076×10^{0}	2.056×10^{-5}	-3.658×10^{-1}	7.172×10^{-1}	-1.120×10^{0}	2.720×10^{-1}
F_{17}	-1.250×10^{1}	3.371×10^{-13}	-9.608×10^{0}	1.622×10^{-10}	-2.962×10^{0}	6.043×10^{-3}	-4.194×10^{0}	2.354×10^{-4}
F_{18}	-1.801×10^{1}	2.752×10^{-17}	-1.777×10^{1}	3.930×10^{-17}	-4.058×10^{0}	3.412×10^{-4}	-4.287×10^{0}	1.828×10^{-4}
F_{19}	-1.886×10^{0}	6.940×10^{-2}	-1.182×10^{0}	2.468×10^{-1}	-4.709×10^{0}	5.685×10^{-5}	-1.200×10^{0}	2.399×10^{-1}
F_{20}	-7.822×10^{0}	1.258×10^{-8}	-2.627×10^{0}	1.363×10^{-2}	-1.677×10^{0}	1.042×10^{-1}	-2.327×10^{0}	2.715×10^{-2}
F_{21}	-2.256×10^{1}	6.096×10^{-20}	-8.681×10^{0}	1.476×10^{-9}	1.939×10^{0}	6.226×10^{-2}	-6.014×10^{-1}	5.523×10^{-1}
F_{22}	-1.092×10^{1}	8.593×10^{-12}	-3.382×10^{0}	2.076×10^{-3}	-4.182×10^{0}	2.434×10^{-4}	-4.385×10^{0}	1.395×10^{-4}
F_{23}	-1.887×10^{1}	7.868×10^{-18}	-2.395×10^{1}	1.174×10^{-20}	-5.113×10^{0}	1.853×10^{-5}	-7.663×10^{0}	1.895×10^{-8}
F_{24}	-1.967×10^{1}	2.579×10^{-18}	-2.128×10^{1}	3.048×10^{-19}	2.003×10^{0}	5.460×10^{-2}	-2.219×10^{0}	3.446×10^{-2}
F_{25}	-1.859×10^{1}	1.176×10^{-17}	-5.979×10^{0}	1.688×10^{-6}	-1.154×10^{1}	2.322×10^{-12}	-8.043×10^{0}	7.194×10^{-9}
F_{26}	-2.662×10^{1}	6.231×10^{-22}	-4.748×10^{0}	5.111×10^{-5}	-6.544×10^{0}	3.618×10^{-7}	-6.443×10^{0}	4.764×10^{-7}
F_{27}	-8.182×10^{0}	5.069×10^{-9}	-1.640×10^{1}	3.285×10^{-16}	1.129×10^{1}	3.900×10^{-12}	1.129×10^{1}	3.902×10^{-12}
F_{28}	-1.764×10^{1}	4.810×10^{-17}	4.099×10^{0}	3.058×10^{-4}	-1.060×10^{1}	1.737×10^{-11}	-9.310×10^{0}	3.254×10^{-10}
F_{29}	-2.047×10^{1}	8.731×10^{-19}	-2.666×10^{1}	6.021×10^{-22}	-6.827×10^{-1}	5.002×10^{-1}	-2.368×10^{0}	2.476×10^{-2}
F_{30}	-1.682×10^{0}	1.033×10^{-1}	-7.369×10^{0}	4.050×10^{-8}	-2.184×10^{0}	3.718×10^{-2}	-1.460×10^{0}	1.550×10^{-1}

Table 4. Wilcoxon signed-rank test results of CEC 2017.

Functions	CS-GWO vs. PSO				CS-GWO vs. WOA			
	p-Value	R+	R−	Winner	*p*-Value	R+	R−	Winner
F_1	1.734×10^{-6}	0	465	+	6.564×10^{-2}	322	143	=
F_2	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_3	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_4	1.734×10^{-6}	0	465	+	2.163×10^{-5}	439	26	−
F_5	1.734×10^{-6}	0	465	+	1.921×10^{-6}	1	464	+
F_6	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_7	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_8	1.734×10^{-6}	0	465	+	2.843×10^{-5}	29	436	+
F_9	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{10}	5.287×10^{-4}	64	401	+	9.627×10^{-4}	393	72	−

Table 4. *Cont.*

Functions	CS-GWO vs. PSO				CS-GWO vs. WOA			
	***p*-Value**	**R+**	**R−**	**Winner**	***p*-Value**	**R+**	**R−**	**Winner**
F_{11}	1.734×10^{-6}	0	465	+	1.238×10^{-5}	20	445	+
F_{12}	1.734×10^{-6}	0	465	+	8.972×10^{-2}	315	150	+
F_{13}	1.734×10^{-6}	0	465	+	1.589×10^{-1}	164	301	=
F_{14}	3.182×10^{-6}	6	459	+	1.150×10^{-4}	420	45	−
F_{15}	1.734×10^{-6}	0	465	+	3.609×10^{-3}	91	374	+
F_{16}	1.734×10^{-6}	0	465	+	5.307×10^{-5}	36	429	+
F_{17}	1.734×10^{-6}	0	465	+	2.879×10^{-6}	5	460	+
F_{18}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{19}	1.734×10^{-6}	0	465	+	1.470×10^{-1}	162	303	=
F_{20}	5.216×10^{-6}	11	454	+	1.480×10^{-2}	114	351	+
F_{21}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{22}	2.353×10^{-6}	3	462	+	3.379×10^{-3}	90	375	+
F_{23}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{24}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{25}	1.734×10^{-6}	0	465	+	1.359×10^{-4}	47	418	+
F_{26}	1.734×10^{-6}	0	465	+	2.613×10^{-4}	55	410	+
F_{27}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{28}	1.734×10^{-6}	0	465	+	6.639×10^{-4}	398	67	−
F_{29}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{30}	1.734×10^{-6}	0	465	+	4.286×10^{-6}	9	456	+
+/=/−				30/0/0				23/3/4

Functions	CS-GWO vs. GWO				CS-GWO vs. CSO			
	***p*-Value**	**R+**	**R−**	**winner**	***p*-Value**	**R+**	**R−**	**winner**
F_1	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_2	3.182×10^{-6}	6	459	+	1.127×10^{-5}	19	446	+
F_3	1.359×10^{-4}	47	418	+	3.327×10^{-2}	129	336	+
F_4	1.921×10^{-6}	1	464	+	3.882×10^{-6}	8	457	+
F_5	4.196×10^{-4}	61	404	+	7.190×10^{-2}	145	320	=
F_6	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_7	1.470×10^{-1}	162	303	=	4.779×10^{-1}	198	267	=
F_8	8.217×10^{-3}	104	361	+	3.872×10^{-2}	132	333	+
F_9	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{10}	1.238×10^{-5}	445	20	−	2.879×10^{-6}	460	5	−
F_{11}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{12}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{13}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{14}	6.892×10^{-5}	39	426	+	7.865×10^{-2}	147	318	=
F_{15}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{16}	3.709×10^{-1}	189	276	=	7.813×10^{-1}	219	246	=
F_{17}	3.065×10^{-4}	57	408	+	1.319×10^{-2}	112	353	+
F_{18}	6.156×10^{-4}	66	399	+	3.589×10^{-4}	59	406	+
F_{19}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{20}	2.564×10^{-2}	124	341	+	1.986×10^{-1}	170	295	=
F_{21}	7.971×10^{-1}	245	220	=	4.950×10^{-2}	328	137	−
F_{22}	6.156×10^{-4}	66	399	+	1.484×10^{-3}	78	387	+
F_{23}	2.879×10^{-6}	5	460	+	3.405×10^{-5}	31	434	+
F_{24}	5.984×10^{-2}	141	324	=	3.872×10^{-2}	333	132	−
F_{25}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{26}	3.112×10^{-5}	30	435	+	4.286×10^{-6}	9	456	+
F_{27}	1.734×10^{-6}	465	0	−	1.734×10^{-6}	465	0	−
F_{28}	1.734×10^{-6}	0	465	+	1.734×10^{-6}	0	465	+
F_{29}	7.190×10^{-2}	145	320	=	8.130×10^{-1}	221	244	=
F_{30}	1.734×10^{-6}	465	0	−	1.734×10^{-6}	465	0	−
+/=/−				22/5/3	+/=/−			19/6/5

Two indicators including t-value and Sig. (2-tailed) can be obtained from PSTT. If the Sig. (2-tailed) is less than 0.05, it can be concluded that the CS-GWO algorithm is better than the compared algorithm. In addition, four indicators including *p*-value, R+, R− and winner can be obtained from WSRT. If the *p*-value is less than 0.05, the null hypothesis can be rejected at 5% significance level. R+ represents a mean error of the CS-GWO algorithm that is higher than that of the compared one. R− represents a mean error of the CS-GWO algorithm that is lower than that of the compared one. Finally, winner indicates whether the CS-GWO algorithm is superior to the compared algorithm, "+" indicates that the CS-GWO algorithm is better than the compared algorithm, "−" indicates that the CS-GWO algorithm is worse than the compared algorithm and "=" indicates that the performance of the two algorithms display no obvious difference.

It can be seen that most of the Sig. (2-tailed) values in Table 3 are less than 0.05, which means that there is an obvious difference between the CS-GWO algorithm and the comparison algorithms, further indicating that the CS-GWO algorithm is superior to the involved algorithms. In Table 4, most of the combinations (*p*-Value, R+, R, winner) are (1.734×10^{-6}, 0, 465, +), revealing that the CS-GWO algorithm outperforms the comparison algorithms. Furthermore, the results of '+/=/−' is 94/14/12, indicating that the CS-GWO algorithm is better than the compared algorithm in 94 out of 120 cases.

From a statistical perspective, it can be concluded that CS-GWO dominates the other compared algorithms in optimization problems with CEC 2017 test functions.

5.4. Case 3: The Effectiveness of the CS-GWO-DA-BiGRU Model

In order to verify the effectiveness of the combination of the crisscross grey wolf optimization algorithm and the DA-BiGRU model in short-term load forecasting, PSO-DA-BiGRU, WOA-DA-BiGRU and GWO-DA-BiGRU models are compared with the combined CS-GWO-DA-BiGRU model in this case. In this subsection, the number of iterations of the swarm intelligence optimization algorithm is uniformly set to 200, and the number of individuals is set to 20.

The experimental results are shown in Table 5 and Figure 8, where Figure 8 is a comparison chart between the prediction results of different models and the real values on 29–31 December 2018.

Table 5. The experiment results of case study 2.

Prediction Model	RMSE/MW	MAE/MW	SMAPE	R^2
DA-BiGRU	29.053	22.897	3.566	0.937
PSO-DA-BiGRU	28.546	22.285	3.519	0.939
WOA-DA-BiGRU	28.209	22.221	3.471	0.941
GWO-DA-BiGRU	27.895	22.162	3.545	0.942
CSO-DA-BiGRU	27.194	21.255	3.347	0.945
CS-GWO-DA-BiGRU	26.144	20.963	3.337	0.949

As shown in Table 2 and Figure 8, the following conclusions can be drawn:

(1) The effectiveness of the swarm intelligence optimization algorithm:

The prediction model combined with the swarm intelligence optimization algorithm has better prediction performance than the single prediction model (i.e., DA-BiGRU). For example, compared with the DA-BiGRU model, the RMSE, MAE and SMAPE values of the PSO-DA-BiGRU and GWO-DA-BiGRU models are reduced by 1.75% and 3.99%, 2.67% and 3.21% and 1.32% and 0.59%, respectively, and the R^2 values are increased by 0.21% and 0.53%, respectively. This is because the weights and the bias of the DA-BiGRU model are optimized by the swarm intelligence optimization algorithm in the initial stage of training, which can effectively avoid the problems of gradient disappearance and gradient explosion, and further improve the accuracy of load forecasting.

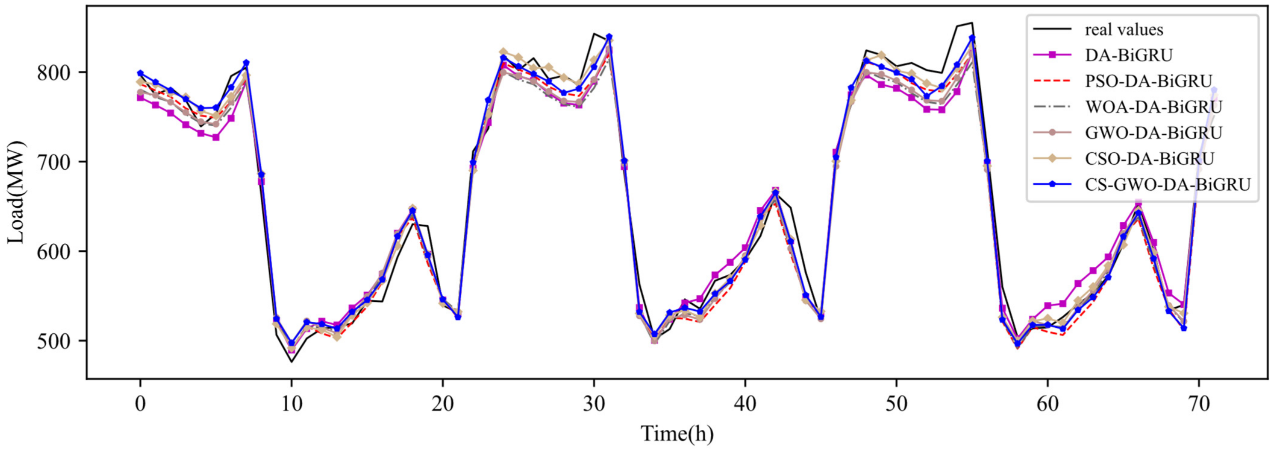

Figure 8. Case study 2: comparison of forecasting results on 29–31 December 2018.

(2) The superiority of the CS-GWO algorithm:

Among all the comparison forecasting models, the proposed CS-GWO-DA-BiGRU short-term load forecasting model has the highest forecasting accuracy. For example, the RMSE, MAE and SMAPE are reduced by 3.86%, 1.37% and 0.30% from those of the second-best performing CSO-DA-BiGRU model, respectively. From the aspect of the R^2 value, the CS-GWO-DA-BiGRU model has an increase of 0.42% compared with the CSO-DA-BiGRU model. Therefore, the CS-GWO algorithm combined with horizontal crossover and vertical crossover operators can improve the global search ability and enhance the diversity of the population, making a great contribution to improving short-term load forecasting.

6. Discussion

In this paper, a high-precision model called CS-GWO-DA-BiGRU is presented in short-term load forecasting problems. However, the proposed model still has some shortcomings that need to be improved. The limitations and future research can be summarized as follows.

(1) The CS-GWO algorithm only focuses on improving the accuracy of short-term load prediction while ignoring the prediction stability, leading to unstable prediction when extending to new data. In the future, we plan to upgrade CS-GWO to a multi-object CS-GWO algorithm to improve the accuracy and stability of short-term load prediction simultaneously.

(2) At present, the intelligent big data platform is valuable for the improvement of the prediction model. In future work, the proposed CS-GWO-DA-BiGRU prediction model will be embedded into the intelligent big data platform to construct an intelligent load forecasting system.

7. Conclusions

Short-term load prediction is essential for the stable operation and safety management of power systems. Therefore, this paper proposes a hybrid model for short-term load prediction, named CS-GWO-DA-BiGRU, which consists of a dual-stage attention mechanism, crisscross grey wolf optimization algorithm and bidirectional gated recurrent unit. The main contributions of this paper can be concluded as follows:

(1) Different from the conventional feature mechanism applied in short-term load forecasting, this paper proposes a dual-stage attention mechanism by combining feature and temporal attention mechanisms. Based on case 1, compared with FA-BiGRU, the RMSE, MAE and SMAPE values of the DA-BiGRU model are reduced by 1.79%, 0.74% and 0.70%, respectively, and the R^2 value is increased by 0.21%. Therefore, DA can effectively capture the correlation relationship of input feature and temporal dependence in load time series simultaneously.

(2) By combining horizontal and vertical crossover operators, the global search ability and population diversity of GWO are enhanced. Based on the Friedman test in case 2, CS-GWO ranks first among the well-known algorithms and achieves the best results for 19 out of 30 functions in CEC 2017. In addition, CS-GWO outperforms the compared algorithms in 94 out of 120 cases based on the Wilcoxon signed-rank test. Furthermore, for the proposed CS-GWO-DA-BiGRU model in case 3, which is based on CS-GWO, the R^2 value has an increase of 0.42% compared with the CSO-DA-BiGRU model and has the best forecasting performance.

Author Contributions: Conceptualization, R.G.; software, X.L.; validation, X.L.; formal analysis, R.G.; investigation, R.G. and X.L.; resources, R.G.; data curation, X.L.; writing—original draft preparation, X.L.; writing—review and editing, R.G.; visualization, X.L.; supervision, R.G.; project administration, R.G.; funding acquisition, R.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China (Grant No. 61561007) and Guangxi Natural Science Foundation (Grant No. 2017GXNSFAA198168).

Data Availability Statement: Not applicable.

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

Nomenclatures

Abbreviations	
WOA	whale optimization algorithm
GWO	grey wolf optimization
LSTM	long short-term memory
GRU	gated recurrent unit
FA	feature attention mechanism
TA	temporal attention mechanism
DA	dual-stage attention mechanism
CS-GWO	crisscross grey wolf optimizer algorithm
RNN	recurrent neural network
BiGRU	bidirectional gated recurrent unit
CSO	crisscross optimization algorithm
HC	horizontal crossover
VC	vertical crossover
RMSE	root mean square error
MAE	mean absolute error
SMAPE	symmetric mean absolute percentage error
R^2	decision coefficient
Mean	mean value
Min	minimum value
Max	maximum value
Std	standard deviation
Rank	ranks
PSTT	paired samples *t*-test
WSRT	Wilcoxon signed-rank test
Formula symbols	
$\boldsymbol{x}_t$	input data at *t*-th time step
$\boldsymbol{h}_{t-1}$,$\boldsymbol{h}_t$,$\boldsymbol{h}_T$	hidden state at ($t-1$)-th, *t*-th and *T*-th time step
$\boldsymbol{r}_t$	reset gate
W_r, U_r, b_r	weight metrices and bias of reset gate
σ	sigmoid activation function
$\boldsymbol{z}_t$	update gate
W_z,U_z,b_z	weight metrices and bias of update gate
W_h,b_h	weight metrices and bias of candidate output
$\widetilde{\boldsymbol{h}}_t$	candidate hidden state

$\overrightarrow{\boldsymbol{h}}_t$,$\overleftarrow{\boldsymbol{h}}_t$	state information of forward and backward propagation
$\overrightarrow{\boldsymbol{w}}_t$,$\overleftarrow{\boldsymbol{w}}_t$	weight metrices of hidden layer in forward and backward propagation
$\boldsymbol{b}_t$	bias of the hidden layer
$G(\cdot)$	calculation process of GRU
$\boldsymbol{e}$	unnormalized attention weight
$\boldsymbol{\alpha}$	normalized attention weight
$\boldsymbol{c}$	intermediate semantic vector
$\boldsymbol{L}^{(i)}$	electric load time series of the previous day
$\boldsymbol{t}^{(i)}$	highest and lowest temperature of the previous day and the current day
$\boldsymbol{r}^{(i)}$	rainfall of the previous day and the current day
$\boldsymbol{d}^{(i)}$	weather day type of the previous day and the current day
$F_{\boldsymbol{\theta}}$	function map of prediction model
$\boldsymbol{X}^{(i)}$	input of prediction model
$\hat{\boldsymbol{Y}}^{(i)}$	predicted load values of the current day
N	number of samples
$\boldsymbol{W}_e$,$\boldsymbol{b}_e$	weight matrix and bias in FA
$\boldsymbol{X}^{\mathrm{ATT}}$	adaptively optimized feature vector
$\boldsymbol{W}_f$,$\boldsymbol{b}_f$	weight matrix and bias in TA
n_p	number of hidden elements in the last layer of BiGRU
$\boldsymbol{d}_t$,$\boldsymbol{d}_T$	intermediate semantic vector in t-th and T-th iteration
$\boldsymbol{W}_y$,$\boldsymbol{b}_y$	weight metrices and bias of the feedforward network in TA
$\boldsymbol{\theta}$	all parameters of DA-BiGRU model
$J(\cdot)$	loss function of DA-BiGRU model
$\boldsymbol{\Phi}$	population of grey wolf
n_M	population size
D	population dimension
$\boldsymbol{\theta}_\alpha(t)$,$\boldsymbol{\theta}_\beta(t)$, $\boldsymbol{\theta}_\delta(t)$,$\boldsymbol{\theta}_\omega(t)$	grey wolves α, β, δ and ω at t-th iteration
A_1,A_2,A_3,C_1, C_2, C_3, A, C	synergy coefficients
r_1, r_2, r_3, r_4, r	random number
$\boldsymbol{\Phi}^{HC}$,$\boldsymbol{\Phi}^{VC}$	offspring population
T	maximum number of iterations
$\boldsymbol{Y}_{test}$,$\hat{\boldsymbol{Y}}_{test}$	actual and predicted load value in testing dataset
$\overline{\boldsymbol{Y}}_{test}$	average value of the actual load value
n_{test}	sample number of the testing dataset

References

1. Vanting, N.B.; Ma, Z.; Jørgensen, B.N. A Scoping Review of Deep Neural Networks for Electric Load Forecasting. *Energy Inform.* **2021**, *4*, 49. [CrossRef]
2. Liu, Y.; Dutta, S.; Kong, A.W.K.; Yeo, C.K. An Image Inpainting Approach to Short-Term Load Forecasting. *IEEE Trans. Power Syst.* **2022**, *38*, 177–187. [CrossRef]
3. Li, L.; Guo, L.; Wang, J.; Peng, H. Short-Term Load Forecasting Based on Spiking Neural P Systems. *Appl. Sci.* **2023**, *13*, 792. [CrossRef]
4. Li, S.; Kong, X.; Yue, L.; Liu, C.; Khan, M.A.; Yang, Z.; Zhang, H. Short-Term Electrical Load Forecasting Using Hybrid Model of Manta Ray Foraging Optimization and Support Vector Regression. *J. Clean. Prod.* **2023**, *388*, 135856. [CrossRef]
5. Mayrink, V.; Hippert, H.S. A Hybrid Method Using Exponential Smoothing and Gradient Boosting for Electrical Short-Term Load Forecasting. In Proceedings of the 2016 IEEE Latin American Conference on Computational Intelligence (LA-CCI), Cartagena, Colombia, 2–4 November 2016; pp. 1–6.
6. Wang, Y.; Kong, Y.; Tang, X.; Chen, X.; Xu, Y.; Chen, J.; Sun, S.; Guo, Y.; Chen, Y. Short-Term Industrial Load Forecasting Based on Ensemble Hidden Markov Model. *IEEE Access* **2020**, *8*, 160858–160870. [CrossRef]
7. Guo, L.; Wu, P.; Lou, S.; Gao, J.; Liu, Y. A Multi-Feature Extraction Technique Based on Principal Component Analysis for Nonlinear Dynamic Process Monitoring. *J. Process Control* **2020**, *85*, 159–172. [CrossRef]
8. Mehdipour Pirbazari, A.; Farmanbar, M.; Chakravorty, A.; Rong, C. Short-Term Load Forecasting Using Smart Meter Data: A Generalization Analysis. *Processes* **2020**, *8*, 484. [CrossRef]
9. Li, B.; Hou, B.; Yu, W.; Lu, X.; Yang, C. Applications of Artificial Intelligence in Intelligent Manufacturing: A Review. *Front. Inf. Technol. Electron. Eng.* **2017**, *18*, 86–96. [CrossRef]

10. Yu, F.; Xu, X. A Short-Term Load Forecasting Model of Natural Gas Based on Optimized Genetic Algorithm and Improved BP Neural Network. *Appl. Energy* **2014**, *134*, 102–113. [CrossRef]
11. Li, S.; Goel, L.; Wang, P. An Ensemble Approach for Short-Term Load Forecasting by Extreme Learning Machine. *Appl. Energy* **2016**, *170*, 22–29. [CrossRef]
12. Lu, H.; Azimi, M.; Iseley, T. Short-Term Load Forecasting of Urban Gas Using a Hybrid Model Based on Improved Fruit Fly Optimization Algorithm and Support Vector Machine. *Energy Rep.* **2019**, *5*, 666–677. [CrossRef]
13. Niu, D.; Dai, S. A Short-Term Load Forecasting Model with a Modified Particle Swarm Optimization Algorithm and Least Squares Support Vector Machine Based on the Denoising Method of Empirical Mode Decomposition and Grey Relational Analysis. *Energies* **2017**, *10*, 408. [CrossRef]
14. Shao, L.; Guo, Q.; Li, C.; Li, J.; Yan, H. Short-Term Load Forecasting Based on EEMD-WOA-LSTM Combination Model. *Appl. Bionics Biomech.* **2022**, *2022*, 2166082. [CrossRef] [PubMed]
15. Li, T.; Qian, Z.; He, T. Short-Term Load Forecasting with Improved CEEMDAN and GWO-Based Multiple Kernel ELM. *Complexity* **2020**, *2020*, 1209547. [CrossRef]
16. Image Segmentation of Leaf Spot Diseases on Maize Using Multi-Stage Cauchy-Enabled Grey Wolf Algorithm. *Eng. Appl. Artif. Intell.* **2022**, *109*, 104653. [CrossRef]
17. Khan, B.; Khalid, R.; Javed, M.U.; Javaid, S.; Ahmed, S.; Javaid, N. Short-Term Load and Price Forecasting Based on Improved Convolutional Neural Network. In Proceedings of the 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), Sukkur, Pakistan, 29–30 January 2020; pp. 1–6.
18. Muzaffar, S.; Afshari, A. Short-Term Load Forecasts Using LSTM Networks. *Energy Procedia* **2019**, *158*, 2922–2927. [CrossRef]
19. Jung, S.; Moon, J.; Park, S.; Hwang, E. An Attention-Based Multilayer GRU Model for Multistep-Ahead Short-Term Load Forecasting. *Sensors* **2021**, *21*, 1639. [CrossRef]
20. Alhussein, M.; Aurangzeb, K.; Haider, S.I. Hybrid CNN-LSTM Model for Short-Term Individual Household Load Forecasting. *IEEE Access* **2020**, *8*, 180544–180557. [CrossRef]
21. Sajjad, M.; Khan, Z.A.; Ullah, A.; Hussain, T.; Ullah, W.; Lee, M.Y.; Baik, S.W. A Novel CNN-GRU-Based Hybrid Approach for Short-Term Residential Load Forecasting. *IEEE Access* **2020**, *8*, 143759–143768. [CrossRef]
22. Cai, C.; Li, Y.; Su, Z.; Zhu, T.; He, Y. Short-Term Electrical Load Forecasting Based on VMD and GRU-TCN Hybrid Network. *Appl. Sci.* **2022**, *12*, 6647. [CrossRef]
23. Alsharekh, M.F.; Habib, S.; Dewi, D.A.; Albattah, W.; Islam, M.; Albahli, S. Improving the Efficiency of Multistep Short-Term Electricity Load Forecasting via R-CNN with ML-LSTM. *Sensors* **2022**, *22*, 6913. [CrossRef]
24. Chen, Q.; Zhang, W.; Zhu, K.; Zhou, D.; Dai, H.; Wu, Q. A Novel Trilinear Deep Residual Network with Self-Adaptive Dropout Method for Short-Term Load Forecasting. *Expert Syst. Appl.* **2021**, *182*, 115272. [CrossRef]
25. Kim, S.H.; Lee, G.; Kwon, G.-Y.; Kim, D.-I.; Shin, Y.-J. Deep Learning Based on Multi-Decomposition for Short-Term Load Forecasting. *Energies* **2018**, *11*, 3433. [CrossRef]
26. Bouktif, S.; Fiaz, A.; Ouni, A.; Serhani, M.A. Optimal Deep Learning Lstm Model for Electric Load Forecasting Using Feature Selection and Genetic Algorithm: Comparison with Machine Learning Approaches. *Energies* **2018**, *11*, 1636. [CrossRef]
27. Kong, X.; Liu, X.; Shi, R.; Lee, K.Y. Wind Speed Prediction Using Reduced Support Vector Machines with Feature Selection. *Neurocomputing* **2015**, *169*, 449–456. [CrossRef]
28. Li, S.; Wang, P.; Goel, L. Wind Power Forecasting Using Neural Network Ensembles with Feature Selection. *IEEE Trans. Sustain. Energy* **2015**, *6*, 1447–1456. [CrossRef]
29. Meng, A.; Chen, S.; Ou, Z.; Ding, W.; Zhou, H.; Fan, J.; Yin, H. A Hybrid Deep Learning Architecture for Wind Power Prediction Based on Bi-Attention Mechanism and Crisscross Optimization. *Energy* **2022**, *238*, 121795. [CrossRef]
30. Zhang, B.; Wu, J.-L.; Chang, P.-C. A Multiple Time Series-Based Recurrent Neural Network for Short-Term Load Forecasting. *Soft Comput.* **2018**, *22*, 4099–4112. [CrossRef]
31. Wang, S.; Wang, X.; Wang, S.; Wang, D. Bi-Directional Long Short-Term Memory Method Based on Attention Mechanism and Rolling Update for Short-Term Load Forecasting. *Int. J. Electr. Power Energy Syst.* **2019**, *109*, 470–479. [CrossRef]
32. Fazlipour, Z.; Mashhour, E.; Joorabian, M. A Deep Model for Short-Term Load Forecasting Applying a Stacked Autoencoder Based on LSTM Supported by a Multi-Stage Attention Mechanism. *Appl. Energy* **2022**, *327*, 120063. [CrossRef]
33. Ribeiro, A.H.; Tiels, K.; Aguirre, L.A.; Schön, T. Beyond Exploding and Vanishing Gradients: Analysing RNN Training Using Attractors and Smoothness. In Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, Online, 26–28 August 2020; pp. 2370–2380.
34. Dewangan, F.; Abdelaziz, A.Y.; Biswal, M. Load Forecasting Models in Smart Grid Using Smart Meter Information: A Review. *Energies* **2023**, *16*, 1404. [CrossRef]
35. Zhang, D.; Kabuka, M.R. Combining Weather Condition Data to Predict Traffic Flow: A GRU-Based Deep Learning Approach. *IET Intell. Transp. Syst.* **2018**, *12*, 578–585. [CrossRef]
36. Xuan, Y.; Si, W.; Zhu, J.; Sun, Z.; Zhao, J.; Xu, M.; Xu, S. Multi-Model Fusion Short-Term Load Forecasting Based on Random Forest Feature Selection and Hybrid Neural Network. *IEEE Access* **2021**, *9*, 69002–69009. [CrossRef]
37. Li, C.; Liu, D.; Wang, M.; Wang, H.; Xu, S. Detection of Outliers in Time Series Power Data Based on Prediction Errors. *Energies* **2023**, *16*, 582. [CrossRef]

38. Niu, Z.; Yu, Z.; Tang, W.; Wu, Q.; Reformat, M. Wind Power Forecasting Using Attention-Based Gated Recurrent Unit Network. *Energy* **2020**, *196*, 117081. [CrossRef]
39. Keskar, N.S.; Socher, R. Improving Generalization Performance by Switching from Adam to Sgd. *arXiv* **2017**, arXiv:171207628.
40. Meng, A.; Zeng, C.; Wang, P.; Chen, D.; Zhou, T.; Zheng, X.; Yin, H. A High-Performance Crisscross Search Based Grey Wolf Optimizer for Solving Optimal Power Flow Problem. *Energy* **2021**, *225*, 120211. [CrossRef]
41. Meng, A.; Chen, Y.; Yin, H.; Chen, S. Crisscross Optimization Algorithm and Its Application. *Knowl. Based Syst.* **2014**, *67*, 218–229. [CrossRef]
42. Guo, Y.; Lu, W.; Li, X.; Huang, Q. Single Image Reflection Removal Based on Residual Attention Mechanism. *Appl. Sci.* **2023**, *13*, 1618. [CrossRef]
43. Mohamed, A.W.; Hadi, A.A.; Fattouh, A.M.; Jambi, K.M. LSHADE with Semi-Parameter Adaptation Hybrid with CMA-ES for Solving CEC 2017 Benchmark Problems. In Proceedings of the 2017 IEEE Congress on Evolutionary Computation (CEC), San Sebastián, Spain, 5–8 June 2017; pp. 145–152.
44. Zhaoyu, P.; Shengzhu, L.; Hong, Z.; Nan, Z. The Application of the Pso Based BP Network in Short-Term Load Forecasting. *Phys. Procedia* **2012**, *24*, 626–632. [CrossRef]
45. Lu, Y.; Wang, G. A Load Forecasting Model Based on Support Vector Regression with Whale Optimization Algorithm. *Multimed. Tools Appl.* **2023**, *82*, 9939–9959. [CrossRef]
46. Barman, M.; Dev Choudhury, N.B. A Similarity Based Hybrid GWO-SVM Method of Power System Load Forecasting for Regional Special Event Days in Anomalous Load Situations in Assam, India. *Sustain. Cities Soc.* **2020**, *61*, 102311. [CrossRef]
47. Meng, A.; Li, Z.; Yin, H.; Chen, S.; Guo, Z. Accelerating Particle Swarm Optimization Using Crisscross Search. *Inf. Sci.* **2016**, *329*, 52–72. [CrossRef]
48. Derrac, J.; García, S.; Molina, D.; Herrera, F. A Practical Tutorial on the Use of Nonparametric Statistical Tests as a Methodology for Comparing Evolutionary and Swarm Intelligence Algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [CrossRef]
49. Hashim, F.A.; Houssein, E.H.; Hussain, K.; Mabrouk, M.S.; Al-Atabany, W. Honey Badger Algorithm: New Metaheuristic Algorithm for Solving Optimization Problems. *Math. Comput. Simul.* **2022**, *192*, 84–110. [CrossRef]
50. Abdel-Basset, M.; Mohamed, R.; Azeem, S.A.A.; Jameel, M.; Abouhawwash, M. Kepler Optimization Algorithm: A New Metaheuristic Algorithm Inspired by Kepler's Laws of Planetary Motion. *Knowl. Based Syst.* **2023**, 110454, *in press*. [CrossRef]
51. Nadimi-Shahraki, M.H.; Zamani, H.; Fatahi, A.; Mirjalili, S. MFO-SFR: An Enhanced Moth-Flame Optimization Algorithm Using an Effective Stagnation Finding and Replacing Strategy. *Mathematics* **2023**, *11*, 862. [CrossRef]
52. Zimmerman, D.W. Teacher's Corner: A Note on Interpretation of the Paired-Samples *t* Test. *J. Educ. Behav. Stat.* **1997**, *22*, 349–360. [CrossRef]
53. Rey, D.; Neuhäuser, M. Wilcoxon-Signed-Rank Test. In *International Encyclopedia of Statistical Science*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 1658–1659.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.