Information Sciences 664 (2024) 120316

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

DIGWO-N-BEATS: An evolutionary time series prediction method for situation prediction

Hao Lin [a,b,c], Chundong Wang [a,b,c,*]

[a] School of Computer Science and Engineering, Tianjin University of Technology, Tianjin, 300384, China
[b] Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology, Tianjin University of Technology, Tianjin 300384, China
[c] Engineering Research Center of Learning-Based Intelligent System, Ministry of Education, Tianjin 300384, China

ARTICLE INFO

Keywords:
Situation awareness
Situation prediction
N-BEATS
Gray wolf optimizer
Parallelization

ABSTRACT

Situation awareness is a key technology in many decision systems, but its performance bottleneck in the third step, namely situation prediction, has not been overcome yet. This step involves time series prediction and commonly uses deep learning methods. However, there are significant performance differences among these methods, and the tuning of their hyperparameters has not been thoroughly investigated. To address these challenges, a novel prediction method named Distributed Improved Gray Wolf Optimizer-Neural Basis Expansion Analysis for Time-Series (DIGWO-N-BEATS) is proposed for situation prediction tasks. First, an architecture based on N-BEATS, which is a cutting-edge paradigm, is formulated for modeling situation value time series. Second, a novel improved evolutionary algorithm, which can converge in parallel, is proposed to optimize thirteen hyperparameters and the model structures of N-BEATS. The experiment results demonstrate that DIGWO-N-BEATS outperforms the six most competitive baselines, reducing the average MAPE on two real-world situation awareness datasets and two time-series prediction datasets by 8.18%, 1.12%, 9.92%, and 4.98%, respectively. Furthermore, DIGWO-N-BEATS exhibits good convergence in hyperparameter optimization tasks and scalability.

1. Introduction

Situation Awareness (SA) is the perception and understanding of the elements in a given environment over a specific time and space, also involves anticipating their future status. SA plays a crucial role in risk assessment and information systems, involving three key levels: perception, comprehension, and projection [1]. The highest level of SA involves the ability to project future states. In this work, our research goal is to achieve higher performance in situation prediction. In previous practice, we have treated these projection tasks as time series prediction tasks, where the situation values (or states) are arranged in a time series and fragmented using the sliding window mechanism [2]. Recently, deep learning has gained significant attention in the realm of time-series prediction and has made significant improvements in situation prediction as well, especially with various recurrent neural networks (RNN) [3–6]. However, the RNN-based prediction models' performance is still unsatisfactory.

N-BEATS, proposed by Oreshkin et al. [7], is a cutting-edge paradigm that has proven excellent nonlinear mapping ability on challenging benchmark datasets of time series prediction, and it is computationally efficient. N-BEATS has achieved State-Of-The-Art (SOTA) results on multiple time series prediction tasks, except for situation prediction tasks. However, despite significant progress,

* Corresponding author.
E-mail address: michael3769@163.com (C. Wang).

https://doi.org/10.1016/j.ins.2024.120316
Received 22 August 2023; Received in revised form 7 February 2024; Accepted 9 February 2024
Available online 15 February 2024

the global optimization of hyperparameters of deep neural networks including N-BEATS remains a challenging problem in deep learning.

Hyperparameter optimization not only reduces human efforts but also improves the performance of deep learning networks [8]. For hyperparameter optimization, the majority of existing deep learning models use grid search (GS) or simple heuristic algorithms in time series prediction or other relevant tasks [9,10]. On the one hand, these optimization algorithms lack sufficient optimization capabilities and explore all potential valuable combinations of the hyperparameters, which can result in poor performance and inefficiency in practice. On the other hand, these optimization algorithms cannot be directly applied to the hyperparameter optimization task of N-BEATS due to its unique parameters, which include both training-hyperparameters and structure-hyperparameters (such as block per stack, stake number, stack type, etc.). Currently, there is a lack of optimization methods to simultaneously optimize training-hyperparameters and structure-hyperparameters of N-BEATS.

In summary, our research motivations are concluded as follows.

- N-BEATS is a cutting-edge paradigm that has been demonstrated with significant improvements for modeling time-series data over RNN-based models, but its feasibility in situation prediction tasks is yet to be proven.
- N-BEATS is expected to perform well in situation prediction, however, the field lacks in-depth investigation of more advanced tuning techniques. No research has yet identified all the hyperparameters requiring optimization for N-BEATS.
- An enhanced time-efficient, robust, and generalizable optimization method is required for the hyperparameter optimization task of N-BEATS.

In this work, we propose an improved evolutionary heuristic algorithm for hyperparameters of N-BEATS to train N-BEATS with a better structure and performance. The proposed heuristic algorithm is termed DIGWO, standing for Distributed Improved Gray Wolf Optimizer, with the purpose of accelerating algorithm efficiency and improving searching ability. Moreover, we propose DIGWO-N-BEATS by combining the two to predict future situations. Our work's main contributions are summarized as follows:

- We used N-BEATS for the first time to achieve situation prediction in this field.
- We have identified all hyperparameters that significantly affect N-BEATS performance. This study is the first to leverage a heuristic algorithm to optimize all thirteen hyper-parameters that affect the performance of N-BEATS.
- We propose a novel Distributed Improved Gray Wolf Optimizer, called DIGWO, which enables us to efficiently discover the unknown hyper-parameter combination of N-BEATS without much prior knowledge. We have developed a new fitness function for N-BEATS that optimizes hyperparameters while maintaining the lightweight structure of N-BEATS.
- We verify the effectiveness and superiority of DIGWO and DIGWO-N-BEATS by extensively testing and comparing them on two situation prediction datasets and two real-world time series datasets.

The remainder of the paper is organized as follows. Section 2 delivers a detailed review of situation prediction research and N-BEATS. Section 3 introduces preliminaries of the situation prediction task and N-BEATS. Section 4 demonstrates the proposed DIGWO-N-BEATS and the proposed key algorithm DIGWO. Section 5 reports experimental results and analysis. Lastly, the paper concludes with a discussion of its limitations and recommendations for future research.

2. Related work

2.1. Situation prediction

In the process of building practical models, situation prediction, which is an important part of situation awareness, is always experimented by situation time series for future situations. This application of machine learning in building situation prediction models has led to a significant improvement in the accuracy of data mining. The reviewed works fall into two categories: the parametric methods and the non-parametric methods. The parametric methods include autoregressive integrated moving average (ARIMA), Grey model, Kalman filtering, etc. However, due to the complex, stochastic, and time-varying relationship between past time series and present time series of situation value, the above parametric methods cannot depict situation value time series precisely with the limited distributional assumptions. The non-parametric methods are mainstream methods, which consisting support vector regression (SVR), radial basis function neural network (RBFNN), back propagation neural network (BPNN), various RNNs, and so on. Table 1 provides an overview of the reviewed methods.

For Network Security Situation Awareness (NSSA), represented by long short-term memory (LSTM), various RNNs are the mainstream prediction models. In order to improve prediction accuracy, Zhao et al. [14] proposed a fusion model with a convolutional neural network (CNN) and LSTM, and used particle swarm optimization (PSO) to optimize the hyperparameters of the fusion model. Some other works have adopted similar improvements. Wang et al. [11] proposed the hybrid hierarchy genetic algorithm (HHGA) to optimize RBFNN, which is used to predict network security situations. Zhang et al. [13] used sparrow search algorithm (SSA) to optimize the weight of BPNN and predicted future cyber security situations. Transformer is a widely used model for time series prediction, proposed in 2017. Yin et al. [18] proposed a new model for predicting network security situations by integrating Temporal Convolutional Network (TCN) with Transformer to enhance its capability to handle nonlinearity. By leveraging the simplicity, autoregressive prediction, and long memory inherent in convolutional architectures for sequential data, the combined model aims to improve the Transformer's predictive performance in network security situations. To clean the situation data, Wen et al. [19]

Table 1
Summary of prediction methods in situation prediction.

Reference	Year	Field	Prediction Method
[17]	2020	Mining safety situation prediction	PSO+ GRU
[11]	2020	Network Security situation awareness	GA+RBF
[12]	2021	Situation awareness in intersection waterways	LSTM
[13]	2022	Network Security situation awareness	SSA+BPNN
[14]	2022	Network Security situation awareness	PSO + ABiLSTM
[4]	2022	Maritime situation awareness and safety	LSTM
[15]	2022	Power system situation awareness	LSTM+MLP
[16]	2022	Pipeline risk situation awareness	GS+SVR
[18]	2022	Network security situation awareness	TCN +Transformer
[6]	2022	CAN bus security situation awareness	SDAE+BI-LSTM
[19]	2023	Network security situation awareness	Gauss+GRU

additionally introduced Gauss kernel density estimation to remove redundant data. They established a gate recurrent unit (GRU) to effectively extract features from the network security situation data and automatically mine the hidden relationships and the changing trend of situations, resulting in future network security situations from an all-around perspective.

For situation awareness of other scenarios, various RNNs remain the mainstream prediction models. Ma et al. [12] proposed an accumulated long short-term memory to capture the complex and variable movements of individual vessels. This technique helps establish connections between different ship motion patterns and intentions, ultimately enhancing situation awareness in intersection waterways. As part of enhancing maritime situation awareness, Billiah et al. [4] used LSTM to encode sequential data in automated identification systems for forecasting future vessel trajectories. To model long-range dependencies between time series, An et al. [15] predicted the transient stability margin based on the LSTM with an attention mechanism to ensure accurate situation awareness of power system operators in terms of transient stability. Chen et al. [6] developed a situation awareness method called SDAE+Bi-LSTM for the CAN bus. They additionally used a stacked denoising auto-encoder (SDAE) to compress the noisy CAN data and a bidirectional LSTM to further capture the periodic features to improve the accuracy of future situation prediction. Several researchers have observed the performance improvements resulting from model hyperparameters. Zhong et al. [16] adopted SVR optimized by the grid search to predict underground pipeline risk situations. Li et al. [17] utilized a PSO-optimized GRU model to predict mine safety conditions and implemented the prediction model in a fog computing facility.

2.2. N-BEATS for time series prediction

N-BEATS is a cutting-edge paradigm that has demonstrated excellent nonlinear mapping ability on various time series prediction tasks including traffic flow prediction, battery life prediction, cryptocurrency value prediction, probabilistic and electricity load forecasting, etc. N-BEATS possesses three desirable properties: interpretable, applicable without modification to target domains, and fast training. Oreshkin et al. [7], authors of N-BEATS, demonstrated that BEATS is effective at solving the mid-term electricity load prediction problem [24]. To alleviate the inexplicability of macroeconomic prediction models, Wang et al. [20] proposed an interpretable purely data-driven approach called EcoForecast for macroeconomic forecasting based on N-BEATS. EcoForecast showcased strong stability and accuracy across various learning scenarios. Ma et al. [21] constructed multiple degradation features instead of relying on a single feature to improve fault prognosis accuracy. They used N-BEATS to predict the future evolution for each feature with high volatility. Su et al. [22] predicted the Solar Cycle 25 using N-BEATS, which was trained and evaluated with 13 months of smoothed monthly total sunspot numbers. Similarly, Anwar et al. [23] utilized N-BEATS for forecasting meteorological solar radiation. Shaikh et al. [25] improved N-BEATS by utilizing a large dataset containing energy consumption data from 169 smart grid customers and incorporated covariates into N-BEATS, addressing the challenges faced by deep learning models when applied to large datasets with profiles of different smart grid customers. The model yielded superior predictions for daily, weekly, and monthly energy consumption. El Majzoub et al. [26] argue that N-BEATS, due to its characteristics, can predict cryptocurrency returns and perform trend and seasonality decomposition on cryptocurrency return data. Wen et al. [27] proposed a deep-learning forecasting model based on N-BEATS to guarantee coverage for probabilistic short-term load forecasting in the smart grid and simultaneously narrow the prediction interval width. In their approach, they also utilized conformal quantile regression to calibrate the constructed prediction interval by using the residuals in a held-out validation. Zhang et al. [8] employed N-BEATS for real-world traffic flow prediction. They also used a differential evolution algorithm to optimize the hyperparameters of N-BEATS, achieving an accuracy of at least 94% and reducing computational expenses by up to 78.90%.

2.3. Optimized N-BEATS

N-BEATS is a new deep learning model proposed in recent years, and its optimization method has not been widely researched. Jatin Bedi et al. [28] proposed a lightweight STOrage workload time series prediction method, which integrates N-BEATS with a multi-input–multi-output windowing strategy to better capture the historical storage variation patterns of the workload data. To

Table 2
Key notations (Excluding notations in Table 3).

Notations	Definition	Notations	Definition
sv	Situation value	SV	Situation value time series
SV'	SV to be predicted	SL	Length of SV to be predicted
Δt	Time interval of timestamp	$series$	Input of block
$\widehat{x_{series}}$	Backcast output of block	$\widehat{result}$	Forecast output of block
FC	Fully connected layer with Relu activation function	$Linear$	Fully connected layer with Linear activation function
θ^f, θ^b	Expansion coefficient	v^b, v^f	Forecast and backcast basis vectors
$g^b_{series}(), g^f_{series}()$	Function that constrains the structure of outputs	y_m, y'_m	True value, Predicted value
ub_i, lb_i	Lower and upper bounds	N^{pop}	Population size
X, X_{GWO}, X_{DIGWO}	Position vector	C, A	Coefficient vector
r_1, r_2, r_3	Random number	λ	Distance control parameter
$MaxIter$	Maximum number of iterations	d	Number of variables
$Near$	Wolve's neighbors	R	Hunting radius of wolve

address low forecast accuracy due to high volatility, Zhang et al. [29] used TCN to extract high-dimensional features from time series data as the inputs of N-BEATS. However, the above two optimization methods have poor generalization.

Li et al. [30] proposed a short-term wind power prediction model based on a TCN to address the limitations of N-BEATS, which only supports univariate input and overlooks the short-term impact of meteorological factors like wind direction. The proposed model optimizes N-BEATS by incorporating multivariate inputs. Specifically, they designed a TCN stack composed of TCN blocks, which enhances the capability of the standard N-BEATS network to handle multivariate inputs. Similarly, Karamchandani et al. [31] expanded the N-BEATS model to include both exogenous and endogenous variables for the prediction of automated guided vehicle deviation.

The above research did not take into account the optimization of N-BEATS hyperparameters, which could lead to notable performance enhancements. Zhang et al. [8] adopted differential evolution (DE) algorithm to optimize nine hyperparameters of N-BEATS. Similarly, Xu et al. [32] adopted SSA to optimize five hyperparameters of N-BEATS to predict the day-ahead electricity price more accurately. Heuristic algorithms are ideal for optimizing N-BEATS hyperparameter problems, which can be classified as NP-hard [33,34]. However, they did not research all the hyperparameters of N-BEATS's structure, and the optimization ability of DE and SSA is poor.

3. Preliminaries

Table 2 provides a key notation table to aid readers in understanding the equations in our methodology.

3.1. Situation prediction tasks

Definition 1. (Situation value time series): A situation value time series is a set of random variables $sv_t, t \in T$, which can be denoted as SV. sv_t indicates the result of the situation assessment distributed based on some univariate probability distribution function. T indicates a set of timestamps with equidistant time intervals Δt.

Definition 2. (Situation prediction): Situation prediction is to estimate future situation values. The set of fully observed situation values at timestamps $T = \{1, 2, ..., t\}$ can be denoted as $SV = \{sv_1, sv_2, ..., sv_t\}$. The objective of situation prediction is to predict the situation value at $\{t+1, t+2, ..., t+SL\}$, denoted by $SV' = \{sv'_{t+1}, sv'_{t+2}, ..., sv'_{t+SL}\}$. SL indicates the length of situation value time series to be predicted.

3.2. N-BEATS

The base architecture of N-BEATS is simple and generic, yet expressive. N-BEATS has two unique model structures, including: block and stack, which are depicted in Fig. 1 [7].

The i-th block accepts its respective input $series$ and outputs two vectors, $\widehat{x_{series}}$ and $\widehat{result}$. For the first block, its respective $series$ is the input of overall N-BEATS. For the remaining blocks, their inputs $series$ are residual outputs of previous blocks.

The operation of a fully connected network of the i-th block can be described as

$$h_{series,1} = FC_{series,1}(x_{series}), \tag{1}$$

$$h_{series,4} = FC_{series,4}(FC_{series,3}(FC_{series,2}(h_{series,1}))), \tag{2}$$

$$\theta^b_{series} = Linear^b_{series}(h_{series,4}), \theta^f_{series} = Linear^f_{series}(h_{series,4}). \tag{3}$$

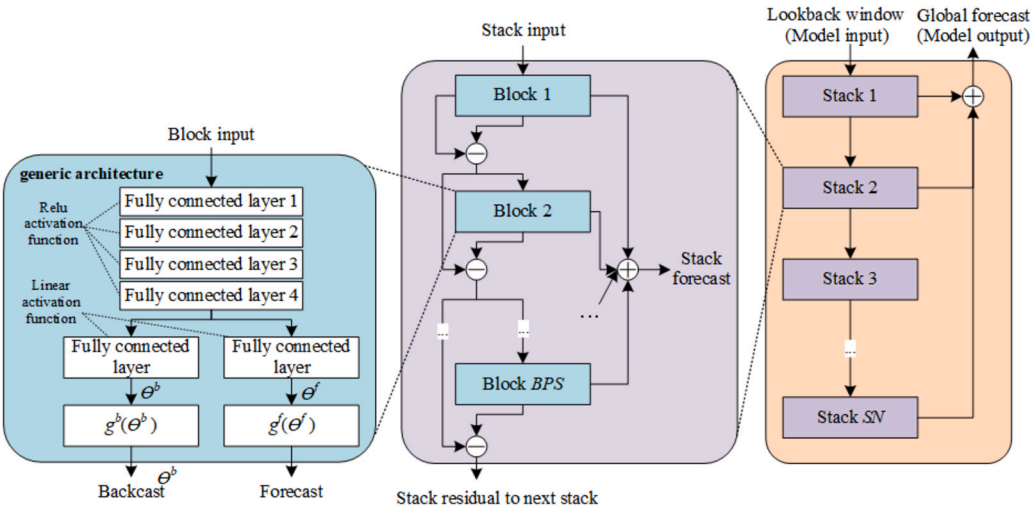

Fig. 1. N-BEATS. It predicts basis expansion coefficients both forward θ^f (forecast) and backward θ^b (backcast). $\{Blocks1, Block2, ..., BlockBPS\}$ are organized into stacks according to the principle of doubly residual stacking. $\{Stacks1, Stack2, ..., StackSN\}$ may have layers with shared g^b and g^f. Finally, the forecasts are hierarchically aggregated.

Then, N-BEATS maps expansion coefficients θ^f and θ^b to outputs via basis layers $\widehat{x_{series}}$ and $\widehat{result}$, which are calculated as

$$\widehat{x_{series}} = g^b_{series}(\theta^b_{series,i}) = \sum_{i=1}^{dim(\theta^b_{series})} \theta^b_{series,i} \cdot v^b_i, \tag{4}$$

$$\widehat{result} = g^f_{series}(\theta^f_{series,i}) = \sum_{i=1}^{dim(\theta^f_{series})} \theta^f_{series,i} \cdot v^f_i. \tag{5}$$

Where, v^b_i and v^f_i are forecast and backcast basis vectors, $\theta^f_{series,i}$ is the i-th element of θ^f_{series}. The function $g^b_{series}()$ and $g^f_{series}()$ is to provide sufficiently rich sets $\{v^b_i\}_{i=1}^{dim(\theta^b_{series})}$ and $\{v^f_i\}_{i=1}^{dim(\theta^f_{series})}$ such that their respective outputs can be represented adequately via varying θ^b_{series} and θ^f_{series}. Each block has three types, including generic block, trend block, and seasonality block, corresponding to different $g^b_{series}()$ and $g^f_{series}()$. The outputs of generic block are described as Eq. (5). For the trend block and seasonality block, the notion of stack is necessary. So, we adopt $\widehat{result}_{s,series}$ to denote the partial forecast of $series$-th block within s-th stack. Trend block and seasonality block are described as

$$\widehat{result}_{s,series} = \begin{cases} \sum_{i=0}^{pd} \theta^f_{s,series,i} \cdot vt^i, series\text{-th block is a trend block} \\ \sum_{i=0}^{\lfloor (SL/2)-1 \rfloor} \theta^f_{s,series,i} cos(2\pi it) + \theta^f_{s,series,i+\lfloor SL/2 \rfloor} sin(2\pi it) \\ \qquad , series\text{-th block is a seasonality block} \end{cases} \tag{6}$$

For the trend block, N-BEATS constrains $g^b_{s,series}()$ and $g^f_{s,series}()$ to be a polynomial of degree pd. And, $vt = [0, 1, ..., SL-2, SL-1]^T / SL$ indicates time vector and is defined on a discrete grid running from 0 to $(SL-1)/SL$, predicting SL steps ahead. For the seasonality block, N-BEATS constrains $g^b_{s,series}()$ and $g^f_{s,series}()$ to be the class of periodic functions, i.e., $\widehat{result}_t = \widehat{result}_{t-\Delta}$, where Δ is a seasonality period. Obviously, different datasets have different the most suitable combinations of blocks.

Oreshkin et al. introduced a new hierarchical doubly residual topology comprising two different residual branches. One branch is responsible for running the backcast prediction of each layer, while the other branch deals with the prediction branch of each layer. It can be described as

$$x_{series} = x_{series-1} - \widehat{x_{series-1}}, \widehat{result} = \sum_{series} \widehat{result_{series}} \tag{7}$$

The structure enables smoother gradient back-propagation by employing a hierarchical decomposition where each block generates a partial prediction. These prediction results $\widehat{result_{series}}$ are aggregated first at the stack-level and then at the overall model-level of N-BEATS.

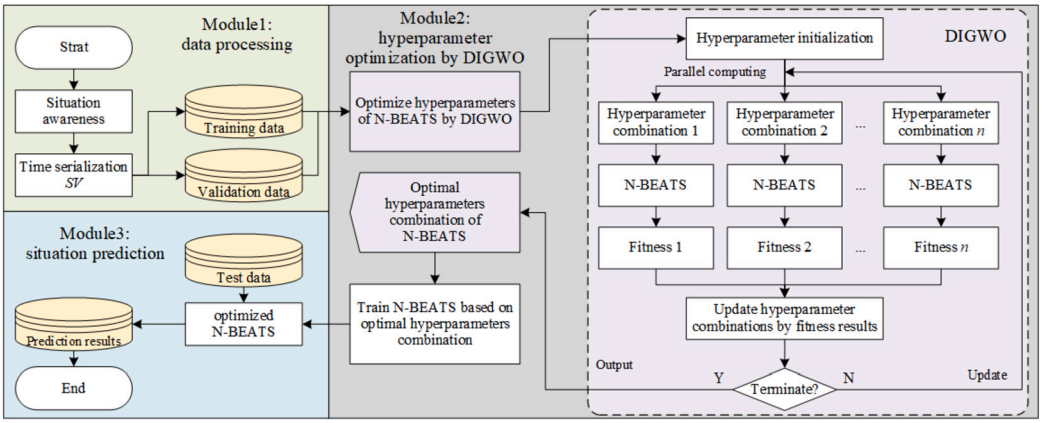

Fig. 2. Workflows of DIGWO-N-BEATS for situation prediction. Where, the complete process of DIGWO is shown in Section 4.4.

Table 3
Summary of hyperparameters that significantly affect the performance of N-BEATS.

Type	Parameter	Range
Model structure hyperparameters	coefficient of SL: $N^{predict}$	$SL \in \{2,3,4,5,6,7\}$
	hidden units: HU	$HU \in [10,500], HU \in \mathbb{Z}^+$
	thetas dim: TD	$TD \in [1,20], TD \in \mathbb{Z}^+$
	stake number: SN	$SN \in [0,8], SN \in \mathbb{Z}^+$
	Block type: BT	$BT \in \{0,1,2\}$
	block per stack: BPS	$BPS \in [1,10], BPS \in \mathbb{Z}^+$
	whether to share all weights within the stack: α_{share}	$\alpha_{share} \in \{0,1\}$
Training hyperparameters	batch size: BS	$BS \in [10,512]$
	learning rate: LR	$LR \in [0.0001, 0.1]$
	weight decay: WD	$WD \in [0,1]$
	exponential decay rate: EDR	$EDR \in [0,1]$
	2nd learning rate decay: EDR'	$EDR' \in [0,1]$

4. Methodology

The workflow of the proposed situation prediction method is depicted in Fig. 2. It contains a data processing module, a hyperparameter optimization module, and a situation prediction module. First, the historical results of situation awareness are time-serialized and divided into a training dataset and a validation dataset. Second, the DIGWO, which forms the core of this paper, was utilized to optimize the hyperparameters of NBEATS through the segmented training and validation datasets. This optimization process is parallelized by Spark. Finally, the future situation values are predicted by the N-BEATS trained with optimized hyperparameters. In the remaining, we will present the details about DIGWO and DIGWO-N-BEATS for situation prediction.

4.1. DIGWO of independent variable (hyperparameters of N-BEATS)

The goal of DIGWO is to find an optimal hyperparameter set that affects the performance of N-BEATS. After extensive analysis and experiments, we have summarized all the hyperparameters that significantly affect the performance of N-BEATS, as shown in Table 3.

We use Adam as an optimizer to train the N-BEATS model. A detailed description of part of the optimized hyperparameters is as follows:

- $N^{predict}$ determines the length of the SV, which is used to predict SV'. Oreshkin et al. [7] proposed an empirical formula as follows.

$$length(SV) = N^{predict} * SL, N^{predict} \in \{2,3,4,5,6,7\} \tag{8}$$

- HU represents the number of hidden units in the first four FC layers (Relu activation) in the block.
- TD represents the number of hidden units in the last two FC layers (Linear activation) in the block.
- SN represents the number of additional stakes in the N-BEATS. To make the solution quality of DIGWO higher, we keep the default value of SN of N-BEATS [7], i.e., a N-BEATS has $SN + 2$ stakes.
- BPS represents the number of blocks in the stakes of N-BEATS. Each stack has at least one block.

- BT determines the type of blocks in the stakes of N-BEATS. To make the solution quality of DIGWO higher, we keep the default value of BT of N-BEATS [7]. By default, a N-BEATS has one stake with BPS trend block and one stake with BPS seasonality block. If $BT = 0$, a N-BEATS has one stake with BPS trend block, one stake with BPS seasonality block, and SN stake with BPS generic blocks; if $BT = 1$, a N-BEATS has $SN + 1$ stake with BPS trend blocks and one stake with BPS seasonality block; if $BT = 2$, a N-BEATS has one stake with BPS trend blocks and $SN + 1$ stake with BPS seasonality block.
- α_{share} determines whether all weights are shared within the stack. α_{share} is a boolean.
- WD determines the decay range of learning rate LR with each epoch. The relevant formula is

$$LR_{i+1} = LR_i * \frac{1}{1 + WD + epoch}, i \in \{1, 2, 3, ..., epoch\} \tag{9}$$

- EDR represents the exponential decay rate for the 1st moment estimates.
- EDR' represents the exponential decay rate for the 2nd moment estimates.

Other hyperparameters are common, so no explanation will be provided.

4.2. DIGWO of dependent variable (fitness function)

The fitness function is the basis for evaluating the solution. The fitness function we proposed consists of two parts: prediction error and structure lightweight, which can be defined as

$$fitness = \frac{1}{SL} \sum_{m=1}^{SL} \| \frac{\tilde{y}_m - y_m}{y_m} \| + (Zoom(SN) + Zoom(BPS)) \tag{10}$$

We convert the multi-objective optimization problem into a new single objective optimization problem through addition. The new optimization problem can be classified as NP-hard. $\tilde{y}$ denote the forecast result of y. We examine the univariate point prediction problem in discrete time. The prediction error is quantified by MAPE, which is a widely used fitness function. The structure lightweight is quantified by $Zoom(SN) + Zoom(BPS)$. $Zoom(p_i)$ refers to zoom the p_i to $[0, 1]$ according to ub_i and lb_i. Since the core advantage of N-BEATS is to use only residual links and fully-connected layers to model time series, we want to keep this lightweight structure as much as possible, so that SN and BPS, which are the hyperparameters most affect model structure complexity are as small as possible without affecting performance. Obviously, SN and BPS will increase n-fold, leading to a proportional increase in the parameter count of the entire N-BEATS.

Additionally, the number of fully connected layers in the block will directly impact the model's lightweight. Nevertheless, the number was pre-determined as 4 by the authors of N-BEATS [7], and we retained the default value without optimization.

4.3. Initial solution generation of DIGWO

In general, at the beginning of the DIGWO, the initial solution needs to be randomly generated in the solution space $Space_i = [ub_i, lb_i]$. However, we found in the results reported in the references and our practice that the SN and BPS of N-BEATS do not need to be too large to achieve good performance [7,8]. So, in order to generate a better initial solution, we utilize the Gaussian Chaotic Map (GCM), which generates random sequences from simple deterministic systems. GCM is employed to generate the initial solutions for SN and BPS, while other parameters are generated randomly as usual. The GCM offers two distinct advantages:

- The GCM can enhance both the evenness and diversity of initial solutions generated [35].
- Compared with the commonly used generating methods of random number and Tent Chaotic Map, the GCM can generate more solutions for $p_i < \frac{ub_i - lb_i}{2}$ and make the initial solution better. This leads to faster convergence of the DIGWO.

The equations below define the Gaussian Chaotic Map.

$$x_{i+1} = \begin{cases} 0 & , x_i = 0 \\ \frac{1}{x_i mod 1} = \frac{1}{x_i} - [\frac{1}{x_i}] & , otherwise. \end{cases} \tag{11}$$

Fig. 3 illustrates the comparison between the initial solution of BPS generated using random numbers and the GCM. The results of SN are similar and will not be displayed again.

4.4. Solution update strategy of DIGWO

DIGWO is a nature-inspired metaheuristic algorithm developed from the grey wolves' internal leadership and group behavior. A group of grey wolves with a population size of N^{pop} are divided into a α wolf (i.e., optimal solution), a β wolf (i.e., second best solution), a γ wolf (i.e., third best solution) and $N^{pop} - 3$ ω wolves by the internal hierarchy. In each iteration, the solution update process of DIGWO is guided by the α, β, and γ wolves. The hyperparameter set of N-BEATS is optimized through simulating grey wolves' surrounding and hunting behavior.

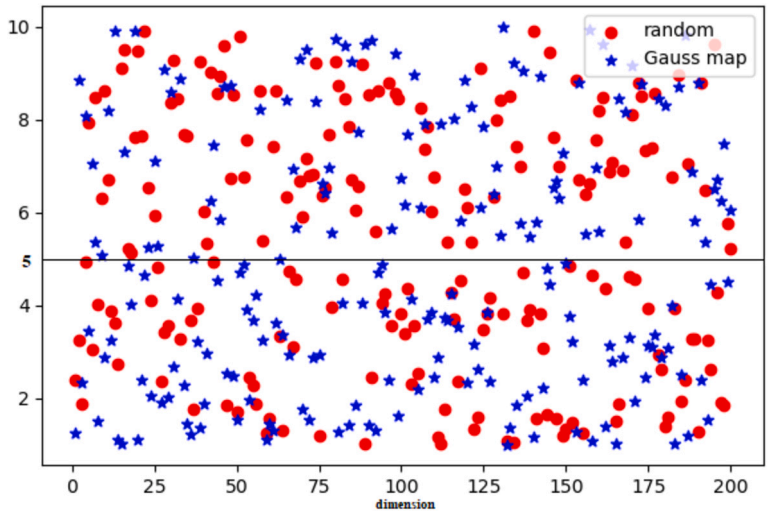

Fig. 3. The comparison between the BPS generated based on random number and Gaussian Chaotic Map. The proportion of red dots below $\frac{ub_i - lb_i}{2} = 5$ is about 50%. The proportion of blue dots below $\frac{ub_i - lb_i}{2} = 5$ is about 55%.

Surrounding the prey by the grey wolves can be modeled as

$$X(t^{GWO}+1) = X_P(t^{GWO}) - A \times D, \tag{12}$$

$$D = | C \times X_P(t^{GWO}) - X(t^{GWO}) | . \tag{13}$$

X is the position vector of a wolf, X_P is the position vector of the surrounded prey, t^{GWO} is the current iteration of DIGWO. The coefficient vectors C and A can be calculated by

$$A = 2\lambda \times r_1 - \lambda, \tag{14}$$

$$C = 2r_2. \tag{15}$$

$$\lambda = 2 - 2t^{GWO}/MaxIter. \tag{16}$$

Where r_1 and r_2 are random numbers, and $r_1, r_2 \in [0, 1]$. λ indicates the distance control parameter, which linearly decreases from 2 to 0 throughout the iterations. $MaxIter$ indicates a maximum number of iterations of DIGWO.

Hunting the prey by the grey wolves can be modeled as

$$X_1(t^{GWO}) = X_\alpha(t^{GWO}) - A_{t^{GWO},1} \times | C_1 \times X_\alpha(t^{GWO}) - X(t^{GWO}) |, \tag{17}$$

$$X_2(t^{GWO}) = X_\beta(t^{GWO}) - A_{t^{GWO},2} \times | C_2 \times X_\beta(t^{GWO}) - X(t^{GWO}) |, \tag{18}$$

$$X_3(t^{GWO}) = X_\delta(t^{GWO}) - A_{t^{GWO},3} \times | C_3 \times X_\delta(t^{GWO}) - X(t^{GWO}) | . \tag{19}$$

Where $X_\alpha(t), X_\beta(t), X_\delta(t)$ have better knowledge about the prey. $A_{t^{GWO},1}, A_{t^{GWO},2}$, and $A_{t^{GWO},3}$ can be calculated by Eq. (14). C_1, C_2, and C_3 can be calculated by Eq. (15).

The behavior of α, β, and γ wolves leading other ω wolves can be modeled as

$$X_{GWO}(t^{GWO}+1) = \frac{X_1(t^{GWO}) + X_2(t^{GWO}) + X_3(t^{GWO})}{3}. \tag{20}$$

However, using Eq. (20) alone will reduce the population diversity of DIGWO in the later search solution process. It leads to DIGWO being prone to falling into local optima. In the real world, grey wolves not only engage in the group search modeled by Eq. (20), but also engage in individual search [36,37]. To enhance population diversity in DIGWO, we simulate individuals receiving hunting information from their neighbors.

Each wolf will learn hunting experience from their neighbors. The position of the i-th wolf in the iteration t^{GWO} is represented as $X_i(t^{GWO}) = \{X_{i,1}, X_{i,2}, ..., X_{i,d}\}$. d indicates the dimension of the objective function, i.e., the number of variables in the problem. The entire wolf group can be recorded as a matrix $Wolves$ with N^{pop} rows and d columns.

The $X_i(t^{GWO})$ wolf's neighbors denoted by $Near_i(t^{GWO})$ can be modeled as

$$\begin{aligned} Near_i(t^{GWO}) = \{ & X_j(t^{GWO}) \mid Euclidean(X_i(t^{GWO}), X_j(t^{GWO})) \le R_i(t^{GWO}), \\ & X_j(t^{GWO}) \in Woloves, j \in [1, 2, ..., d] \}. \end{aligned} \tag{21}$$

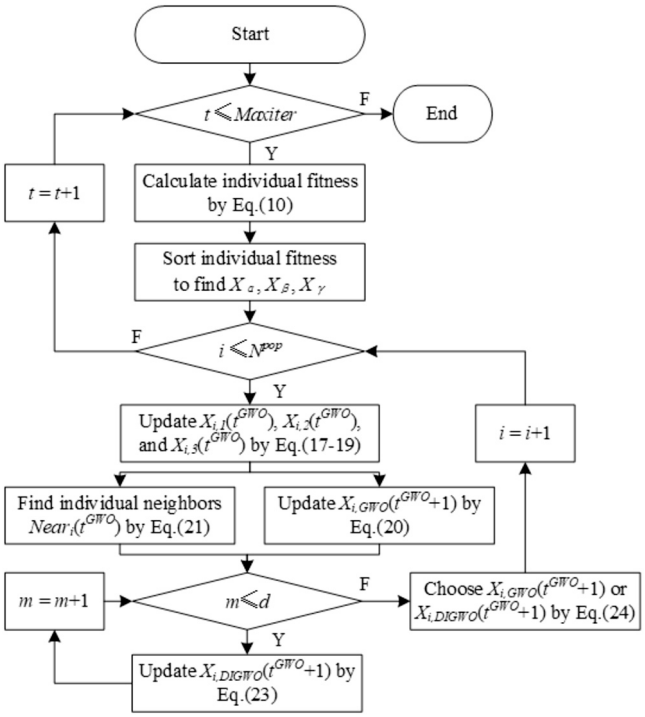

Fig. 4. Process of the solution update in DIGWO.

Where $Euclidean(X_i(t^{GWO}), X_j(t^{GWO}))$ is Euclidean Distance between $X_i(t^{GWO})$ and $X_j(t^{GWO})$. Other distance measurement methods can be selected based on the problem, such as Cosine Similarity and Manhattan Distance. $R_i(t^{GWO})$ indicates the hunting radius of i-th wolf. $R_i(t^{GWO})$ can be defined as

$$R_i(t^{GWO}) = Euclidean(X_i(t^{GWO}), X_{i,GWO}(t^{GWO}+1)). \tag{22}$$

Where $X_{i,GWO}(t^{GWO}+1)$ represents the result of the group search, which can be calculated by Eq. (20). Similarly, the Euclidean Distance here can be replaced. The result $X_{i,DIGWO}$ of the individual search can be defined as

$$\begin{aligned} X_{i,DIGWO}(t^{GWO}+1) &= X_i(t^{GWO}) + r_3 \times (X_n(t^{GWO}) - X_r(t^{GWO})), \\ X_n(t^{GWO}) &\in Near_i(t^{GWO}), X_r(t^{GWO}) \in Woloves. \end{aligned} \tag{23}$$

Where $X_n(t^{GWO})$ is randomly sampled from $Near_i(t^{GWO})$ and $X_r(t^{GWO})$ is randomly sampled from $Woloves$. And, r_3 is a random number, and $r_3 \in (0, 1]$.

Finally, $X_{i,DIGWO}(t^{GWO}+1)$ should be compared with $X_{i,GWO}(t^{GWO}+1)$, i.e.,

$$X_i(t^{GWO}+1) = \begin{cases} X_{i,GWO}(t^{GWO}+1), & fitness(X_{i,GWO}) < fitness(X_{i,DIGWO}) \\ X_{i,DIGWO}(t^{GWO}+1), & fitness(X_{i,GWO}) \geq fitness(X_{i,DIGWO}). \end{cases} \tag{24}$$

Upon completing the individual search for all wolves, t^{GWO} increase by 1.

The process of the solution update in DIGWO is shown in Fig. 4.

4.5. Parallelization of DIGWO

While DIGWO can provide a satisfactory solution within a limited timeframe, there is room for significant improvement in terms of efficiency. To improve the efficiency of DIGWO, we propose a parallelized design for DIGWO based on Spark, an engine that executes data engineering, data science, and machine learning on clusters.

The current parallel design of heuristics algorithms involves parallelizing all essential steps, such as updating solutions (e.g., mutation in GA and updating wolf position in GWO) and computing the fitness function [38,39]. However, the existing design is not suitable for our DIGWO. We found that the majority of DIGWO's running time is attributed to the computation time of the fitness function, since every time Eq. (10) is calculated, N-BEATS needs to be trained once. Moreover, as N^{pop} increases, the calculation

time of Eq. (10) becomes larger. When a parallelized algorithm is launched, the Spark cluster requires time to perform fundamental operations like initiating jobs, task allocation, and resource allocation. If the computation performed by the Executor in the Spark cluster is minimal, the parallelization performance of the cluster is adversely affected. So, we exclusively design parallel DIGWO solely for the fitness function, i.e., the calculation of Eq. (10). The pseudocode of parallelized DIGWO is shown in Algorithm 1.

Algorithm 1 Parallelized DIGWO.

Input: N^{pop}, $MaxIter$, Spark cluster configuration parameters $conf$.
Output: Optimal parameter set $P_{best} = \{p_1, p_2, p_3, ..., p_{12}\}$

1: Create SparkContext object sc using $conf$ //**Initialization phase**
2: Initialize solutions pop using N^{pop} according to Eq. (11).
3: Creat RDD data RDD^{pop} using pop by sc.parallelize().
4: Compute the fitness of each solution in RDD^{pop} in parallel by RDD^{pop}.map(getFitness()). //**Computing the fitness function**
5: Return the fitness value in RDD^{pop} to list $fitness$ by RDD^{pop}.collect().
6: $t^{GWO} = 0$
7: **while** $t^{GWO} \geq MaxIter$ **do** //**Updating solutions**
8: Find α wolf, β wolf, and γ wolf according to $fitness$.
9: Update pop according to Eq. (12)-(24).
10: Create RDD data RDD^{pop} using pop by sc.parallelize().
11: Compute the fitness of each solution in RDD^{pop} in parallel by RDD^{pop}.map(getFitness()). //**Computing the fitness function**
12: Return the fitness value in RDD^{pop} to list $fitness$ by RDD^{pop}.collect().
13: $fitness_{best}$ = Min($fitness$).
14: tmp = tmp + 1
15: **end while**
16: Decode $fitness_{best}$ to get optimal parameter set $P_{best} = \{p_1, p_2, p_3, ..., p_{13}\}$
17: **Return** P_{best} **and End algorithm**

The purpose of Algorithm 1 is to convert the solutions in DIGWO into Resilient Distributed Dataset (RDD) data format using the parallelize() function, which is a distributed memory abstraction that allows for the evaluation of the solution set in parallel by using the map() function to calculate solution's fitness on large clusters. The collect() function can trigger the transformation operations, such as map(), which are not executed due to Spark's lazy computing characteristics. Once $MaxIter$ is reached, the DIGWO returns the optimal parameter set that was found. The map() function on lines 4 and 11 is the core of the DIGWO as it processes each element in the RDD^{pop} using a specified getFitness() function, resulting in a new RDD data. Algorithm 2 displays the pseudocode of the getFitness() function.

Algorithm 2 *getFitness*().

Input: Train data $data_t$, Validation data $data_v$, RDD^{pop}
Output: $fitness_i$ corresponding to $solution_i$

1: Decode the N^{pop} parameter sets of N-BEATS according to $solution_i$ in RDD^{pop}. Store the parameter sets in Set.
2: **for** i **in** Set **do** //i is a set of N-BEATS parameters $\{p_1, p_2, p_3, ..., p_{12}\}$
3: Train N-BEATS using i and $data_t$.
4: Calculate error $mape_i$ of N-BEATS using $data_v$.
5: Calculate $fitness_i$ using $mape_i$ and i by Eq. (10).
6: **end for**
7: **Ruturn** $fitness_i$ **and End algorithm**

4.6. Convergence proof of DIGWO

Since DIGWO is a complex improved algorithm derived from converged GWO, we adopt the related theorems of Markov chain to prove the convergence of DIGWO, which are widely used for convergence analysis of heuristic algorithms. Firstly, we construct a Markov model for DIGWO.

Definition 3. In DIGWO, the state of the grey wolves is denoted as ξ. The Markov state space of the wolves is denoted as $\Xi = \{\xi \mid \xi \in Z\}$. Where Z is solution space of DIGWO. The state of the wolf groups is denoted as $\phi = (\xi_1, \xi_2, ..., \xi_i), i = 1, 2, ..., N^{pop}$. The Markov state space of the wolf groups is denoted as $\Phi = \{\phi = (\xi_1, \xi_2, ..., \xi_i) \mid \xi_i \in \Xi, i = 1, 2, ..., N^{pop}\}$.

Definition 4. In DIGWO, $\forall \xi_i, \xi_j \in \Xi, i, j = 1, 2, ..., N^{pop}$, the state transition of the wolves is denoted as $Transition_{\phi}(\xi_i) = \xi_j$. $\forall \phi_i, \phi_j \in \Phi$, the state transition of the wolf groups is denoted as $Transition_{\Phi}(\phi_i) = \phi_j$. The transition probability of the wolf groups is denoted as

$$P(Transition_{\Phi}(\phi_i) = \phi_j) = \prod_{m=1}^{N^{pop}} P(Transition_{\phi}(\xi_{im}) = \xi_{jm}). \tag{25}$$

Then, from previous research, we can see that

Theorem 1. *The Markov state sequence of wolf groups in GWO is a finite homogeneous Markov chain [40].*

Theorem 2. *According to the convergence criterion [40], if the Markov state sequence of the wolf groups is a finite homogeneous Markov chain, GWO is convergent [41].*

Hence, we need to prove

Theorem 3. *The Markov state sequence of wolf groups in DIGWO is a finite homogeneous Markov chain.*

Next, we prove Theorem 3.

Proof. The **finiteness**, **Markov property**, and **homogeneity** of the state sequence of the wolf groups in DIGWO must be proved.

First, the Markov search space of any optimization algorithm is limited [40]. So, the Markov search space of solution of DIGWO is finite, ξ_i is finite, and Ξ is finite. Since, Φ consists of Ξ_i, the number of Ξ_i is finite positive integer, so Φ is finite, and then the Markov state sequence of the wolf groups $\{\phi(t^{GWO}) \mid t^{GWO} > 0\}]$ has **finiteness**.

Based on Definition 4, $\forall \phi(t^{GWO}-1), \phi(t^{GWO}) \in \Phi$, the transition probability $P(Transition_{\Phi}(\phi(t^{GWO}-1)) = \phi(t^{GWO}))$ of the Markov state sequence of the wolves $\{\phi(t^{GWO}) \mid t^{GWO} > 0\}]$ is determined by the transition probability $P(Transition_{\Phi}(\xi(t^{GWO}-1)) = \xi(t^{GWO}))$ of wolves in the wolf groups. According to Eq.(12-24), $P(Transition_{\Phi}(\xi(t^{GWO}-1)) = \xi(t^{GWO}))$ is only related to $\Phi(\xi(t^{GWO}-1))$, D and r_3. So, $P(Transition_{\Phi}(\phi(t^{GWO}-1)) = \phi(t^{GWO}))$ is only related to the state at $t^{GWO}-1$. Based on the fundamental properties of a Markov chain, $\{\phi(t^{GWO}) \mid t^{GWO} > 0\}]$ has **Markov property**.

A Markov chain is considered homogeneous if the one-time-step transition probability of the Markov state sequence is independent of the starting time [40]. $P(Transition_{\Phi}(\xi(t^{GWO}-1)) = \xi(t^{GWO}))$ is only related to the state at $t^{GWO}-1$ and unrelated to $t^{GWO}-1$. So, $\{\phi(t^{GWO}) \mid t^{GWO} > 0\}]$ has **homogeneity**.

Hence, the Markov state sequence of the grey groups in the DIGWO is a finite homogeneous Markov chain.

In summary, the DIGWO is convergent.

5. Experiment

5.1. Research question

The analysis carried out in our experiments is aimed at investigating three main research questions (RQ):

RQ1 Is DIGWO the most suitable heuristic algorithm for optimizing N-BEATS hyperparameters?
RQ2 Do each improvement of the DIWO-N-BNEATS we designed achieve the expected results?
RQ3 Do DIGWO-N-BNEATS improve efficiency by using parallelization?

The following research is organized into four tasks. In the first one, we trialed the performance of multiple heuristic algorithms for optimizing N-BEATS hyperparameters (for RQ1). In the second one, we conducted ablation experiments on both the GCM and the individual search (for RQ2). In the third one, we measured the scalability of DIGWO in clusters with different configurations (for RQ3). In the last one, we compared DIGWO-N-BNEATS with other existing methods for situation prediction.

5.2. Data

Two real-world situation prediction datasets were collected for evaluating DIGWO-N-BEATS. Cncert dataset originates from the number of computer malicious program propagation times, websites tampered with, backdoor websites implanted, counterfeit websites, and new information security vulnerabilities published by the National Internet Emergency Center, from July 31, 2017 to March 14, 2021.[1] The weights of the five security threats are $w = [0.3, 0.25, 0.15, 0.15, 0.15]$. Covid-Situation contains COVID-19 situation data for a certain region, from 04 April, 2020 to 20 December, 2021, specifically the number of lab tests, the number of confirmed cases, and death cases.[2] The weights of the three elements are $w = [0.2, 0.3, 0.5]$. The situation value in Cncert and Covid-Situation can be calculated as

$$situation \quad value = \sum_{i=1} \frac{situation \quad element_i}{max(situation \quad element)_i} \cdot w_i \tag{26}$$

Due to the small size of existing situation awareness datasets, and the applicability of DIGWO-N-BEATS to any time series prediction task, we choose three real-world datasets to further evaluate the performance. Milk dataset includes monthly milk production

[1] https://www.cert.org.cn.
[2] https://github.com/TanvirJoardar/Bangladesh-Covid-Situation-Analysis-and Trackin.

Table 4
Dataset description.

Dataset	Description	Number of sample	Time interval Δt	SL	Epoch
Cncert	the number of five security threats	190	week	5	20
Covid-Situation	the number of three covid elements	626	day	5	40
Milk	monthly milk production	169	month	5	20
Sunspot	monthly number of sunspots	3651	day	5	50

Table 5
Results of multiple heuristic algorithms for optimizing N-BEATS.

Algorithm	Cncert			Covid-Situation		
	MAE	MAPE	Fitness	MAE	MAPE	Fitness
EGA	3.13	10.20	10.23	0.81	11.07	11.32
DE	3.20	10.49	11.12	0.88	11.29	11.35
GSA	3.43	10.78	11.20	1.06	13.63	14.05
GWO	3.13	10.24	10.58	0.85	10.92	11.09
MVO	3.68	11.24	11.70	0.83	10.72	11.03
SSA	3.02	9.68	10.74	0.85	11.30	11.40
SCA	3.25	10.07	10.07	0.88	11.45	11.47
BBO	3.04	9.65	9.94	0.81	10.90	10.91
SPBO	3.18	10.47	11.03	0.92	12.30	12.69
DIGWO	**2.92**	**9.53**	**9.56**	**0.79**	**10.691**	**10.695**

from January 1962 to December 1975.[3] Sunspots dataset includes monthly number of sunspots from January 1749 to December 1983.[4] Table 4 presents information on the four datasets. The ratio of training set size to test set size is 8:2.

We divide the test set into N_{sample} sets of input and output results, $N_{sample} = Numberof\,sample * 0.2 - N^{predict} - SL + 1$. We consider MAE average and MAPE average as evaluation indicators.

$$MAE = \frac{1}{N_{sample}} \sum_{i=1}^{N_{sample}} \frac{1}{SL} \sum_{m=1}^{SL} ||\tilde{y}_m^i - y_m^i|| \tag{27}$$

$$MAPE = \frac{1}{N_{sample}} \sum_{i=1}^{N_{sample}} \frac{1}{SL} \sum_{m=1}^{SL} ||\frac{\tilde{y}_m^i - y_m^i}{y_m^i}|| \tag{28}$$

5.3. Convergence ability experiment

The No Free Lunch (NFL) theorem [42] prevents us from directly selecting optimization algorithms for N-BEATS hyperparameters. Therefore, we compare DIGWO with other heuristic algorithms to demonstrate the superiority of DIGWO in optimizing N-BEATS hyperparameters. The heuristic algorithms include: Genetic Algorithm with elitist strategy (EGA), Differential Evolution (DE), Gravitational Search Algorithm (GSA) [43], GWO, Multi-Verse Optimizer (MVO) [44], SSA [45], Sine Cosine Algorithm (SCA) [46], Biogeography Based Optimization (BBO) [47], and Student psychology based optimization (SPBO) [48].

N^{pop} of DIGWO is set to 10. In order to balance the convergence ability of algorithms, N^{pop} of other heuristic algorithms is set to 20. $MaxIter$ of all heuristic algorithms is set to 20, which is sufficient for convergence of these heuristic algorithms. In practical applications, it is advisable to maximize N^{pop} and $MaxIter$ within time constraints to achieve the best optimization results. Other parameters of above heuristic algorithms include: 0.1 mutation probability of EGA; 0.5 crossing probability of EGA; 0.9 scale factor of DE; 0.2 crossing probability of DE; wormhole existence probability of MVO increases linearly from 0.2 to 1; shuttle distance ratio of MVO decreases from 0.6 to 0; decrease coefficient of SSA decreases linearly from 2 to 0; 2 control parameters of SCA; 0.1 mutation probability of BBO; λ of GWO and DIGWO decreases linearly from 2 to 0. The experimental results are shown in Table 5.

As shown in Table 5, the best fitness achieved by DIGWO beats the second-best fitness achieved by 1.98% in the Cncert dataset. Similarly, in the Covid-Situation dataset, the best fitness achieved by DIGWO beats the second-best fitness achieved by 2.01%. Moreover, the MAE and MAPE obtained by DIGWO are the lowest in both datasets. This indicates that DIGWO outperforms other algorithms in terms of convergence ability. Figs. 5 and 6 display the fitness convergence curves of the optimal outcomes from each heuristic algorithm.

[3] https://raw.githubusercontent.com/jbrownlee/Datasets/master/monthly-sunspots.csv.
[4] https://raw.githubusercontent.com/jbrownlee/Datasets/master/monthly-sunspots.csv.

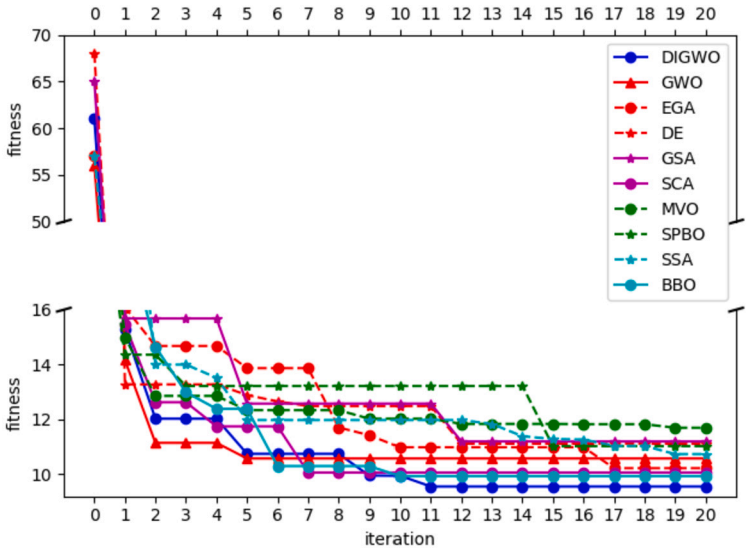

Fig. 5. Optimal fitness convergence curves of heuristic algorithms on Cncert dataset. The hyperparameter set of DIGWO is $N^{predict} = 2$, $HU = 109$, $TD = 8$, $SN = 0$, $BT = 2$, $BPS = 1$, $\alpha_{share} = 0$, $BS = 276$, $LR = 0.0472$, $WD = 0.9008$, $EDR = 0.0662$, $EDR' = 0.2857$.

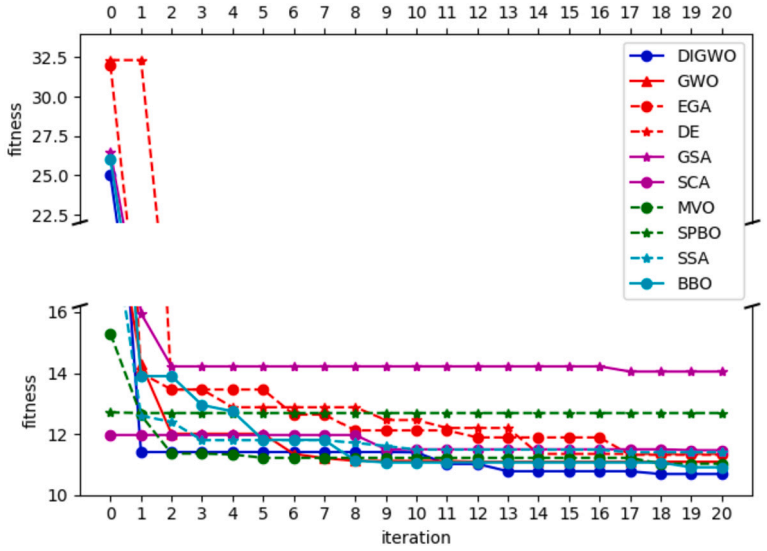

Fig. 6. Optimal fitness convergence curves of heuristic algorithms on Covid-Situation dataset. The hyperparameter set of DIGWO is $N^{predict} = 5$, $HU = 439$, $TD = 1$, $SN = 0$, $BT = 1$, $BPS = 1$, $\alpha_{share} = 0$, $BS = 13$, $LR = 0.0702$, $WD = 0.7913$, $EDR = 0.2524$, $EDR' = 0.2801$.

5.4. Ablation experiment

The purpose of the improvements-level ablation experiment is to prove the necessity of our improvements for DIGWO. These improvements include:

- G: Gaussian Chaotic Mapping
- I: Individual Search Strategy

N^{pop} and $MaxIter$ of DIGWO is set to 20. The experiment results are shown in Table 6.

The experiment results demonstrate that 'GWO+G+I' obtained the lowest MAE, MAPE, and fitness on both situation awareness datasets. Compared to the GCM, the individual search strategy has a greater impact on the performance of DIGWO, and both gaussian chaotic mapping and individual search strategy are crucial for optimal results.

Furthermore, it was discovered that for small datasets like Milk, the values of SN and BPS in the optimal hyperparameter set consistently exhibit very low levels. To address this issue, we recommend increasing the weight of prediction errors within the fitness function during the implementation of DIGWO-N-BEATS on small datasets.

Table 6
Ablation experiment results for DIGWO.

Improvement	Cncert			Covid-Situation		
	MAE	MAPE	Fitness	MAE	MAPE	Fitness
GWO+G	3.01	9.6	9.61	0.83	10.78	10.79
GWO+I	2.98	9.56	9.59	0.81	10.74	10.75
GWO+G+I	**2.92**	**9.53**	**9.56**	**0.79**	**10.691**	**10.695**

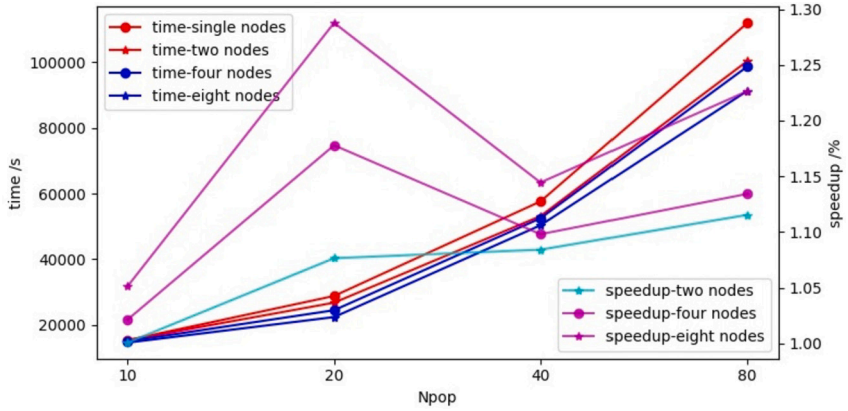

Fig. 7. Time cost and speedup of DIGWO under different circumstances on Cncert dataset.

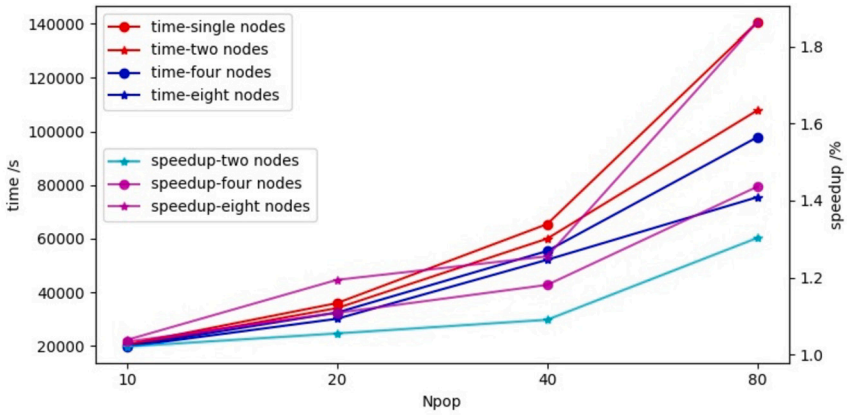

Fig. 8. Time cost and speedup of DIGWO under different circumstances on Covid-Situation dataset.

5.5. Scalability experiment

DIGWO is implemented on different configurations of clusters, including a single node (stand-alone), 2 nodes, 4 nodes, and 8 nodes, to record the time-consuming and the speedup of the algorithm. We conducted the above experiments using Beijing ChinaHPC Technology Co., Ltd.'s high-performance computing cluster.

The specific parameters for Spark are executor.pyspark.memory = 2G; executor.cores = 1; num-executors = 4; driver.cores = 1; driver.memory = 1G; and python.worker.memory = 1G. These parameters are determined by server performance and do not impact the experiment conclusions. The experimental datasets used in the experiment are Cncert and Covid-Situation. Speedup is defined as follows:

$$Speedup = \frac{time_1}{time_n} \times 100\% \tag{29}$$

Where, $time_1$ is the sequential calculation time. $time_n$ is parallel calculation time based on n nodes. Ideally, the speedup is equal to the number of cluster nodes. The experimental results are shown in Figs. 7 and 8.

As shown in Figs. 7 and 8, the time cost of DIGWO gradually increases with the gradual increase of the calculation. Initially, when the calculation is less, the speedup of DIGWO is low. This can be attributed to the low acceleration of the Spark cluster, which is mainly due to the basic operations involved. However, as the calculation increases gradually, the advantages of the proposed parallel design become more and more apparent. Consequently, the speedup of DIGWO gradually approaches closer to the ideal value.

Table 7
The comparison between SOTA research and our DIGWO-N-BEATS.

Dataset	Metric	GS+SVR	LSTM+MLP	Gauss+GRU	TCN+TX	N-BEATS	ARDE-N-BEATS	DIGWO-N-BEATS
cncert	MAE	3.26	4.89	4.99	4.09	5.34	3.22	**2.92**
	MAPE	10.53	14.33	14.73	12.62	16.3	10.31	**9.53**
Covid	MAE	1.86	1.45	1.69	0.87	1.03	0.81	**0.79**
	MAPE	25.77	18.79	19.33	11.04	14.05	10.81	**10.69**
milk	MAE	5558	5315	5427	4248	5180	4037	**3623**
	MAPE	6.89	6.26	6.38	4.96	5.98	4.65	**4.23**
sunspot	MAE	301.1	289.8	293.5	221.3	243.6	228.5	**203.7**
	MAPE	31.45	28.47	29.04	21.69	24.03	22.13	**20.66**

Our proposed parallelization method does not harm the training data, hence it does not diminish the performance of DIGWO. In addition, we found two outliers at $N^{pop} = 20$. This can be attributed to SN, BPS, and BS significantly affecting the time consumption of DIGWO, while DIGWO randomly selects these parameters during the iteration process. In summary, the experiment results verify that DIGWO has good scalability.

5.6. Compare with the SOTA methods

To verify the performance of our proposed method, we compare it to the current state-of-the-art situation prediction methods on the above four datasets. These situation prediction methods consist of the following:

- GS+SVR represents a SVR optimized by grid search for hyper-parameters [16].
- LSTM+MLP represents a model consisting of a two-layer LSTM and a two-layer MLP. The attention mechanism is applied after the first layer of the LSTM [15].
- Gauss+GRU represents a model consisting of a two-layer GRU, replacing Softmax output with regression output. The redundant training data is cleaned by Gauss kernel density estimation [19].
- TCN+TX represents a model consisting of a Temporal Convolutional Network-Based input embedding module and a Transformer [18].
- N-BEATS represents a N-BEATS using default parameters [7].
- ARDE-N-BEATS represents a N-BEATS optimized by DE for hyper-parameters except for BT and α_{share} [8].

We first prioritize citing the experiment results mentioned in the above literature. If these results are not available, we recreate their methods using the hyperparameters that we have optimized on the above four datasets. The comparison between our research and existing SOTA research is presented in Table 7.

Obviously, our proposed DIGWO-N-BEATS clearly outperformed on all four datasets. DIGWO-N-BEATS outperforms the four most competitive baselines, reducing the average MAPE on four datasets by 8.18%, 1.12%, 9.92%, and 4.98%, respectively, and reducing the average MAE by 10.27%, 2.53%, 11.42%, and 8.64%, respectively. DIGWO-N-BEATS, leveraging the inherent advantages of N-BEATS, outperforms GS+SVR, LSTM+MLP, and Gauss+GRU by a considerable margin. The superior performance of DIGWO, compared to ARDE, highlights the impact of the BT and α_{share} we introduced on the performance of N-BEATS, making their optimization necessary. In summary, the experiment results verify that DIGWO-N-BEATS is the SOTA method for situation prediction tasks.

6. Conclusion

This work proposed a novel framework, DIGWO-N-BEATS, for situation prediction tasks. The main idea of this work was to utilize an enhanced GWO method, called DIGWO, to globally optimize the hyperparameters of N-BEATS. The DIGWO-N-BEATS algorithm's superiority was confirmed through the evaluation on four datasets.

There are still some limitations to our work, including:

- We only optimized the type and number of blocks in N-BEATS without modifying its block structure. It is possible that the current block structure is not optimal.
- This work cannot be directly applied to N-BEATSX [49] due to its unique hyperparameters.
- In this work, the situation prediction task is modeled as a time-series prediction problem. However, various external factors, such as months and regions, need to be taken into account in the situation prediction task.

In future work, we aim to refine our DIGWO by investigating external factors, to optimize N-BEATSX for situation prediction tasks and achieve more accurate situation awareness. Furthermore, we will combine parallelization with N-BEATS to achieve more efficiency.

CRediT authorship contribution statement

Hao Lin: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Chundong Wang:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Chundong Wang reports financial support was provided by National Natural Science Foundation of China No. U1536122. Chundong Wang reports financial support was provided by Ministry of Science and Technology of the People's Republic of China No. SQ2020YFF0413781. Chundong Wang reports financial support was provided by Ministry of Industry and Information Technology of the People's Republic of China.

Data availability

Data will be made available on request.

Acknowledgements

This work was supported by National Natural Science Foundation of China No. U1536122, Ministry of Science and Technology of the People's Republic of China No. SQ2020YFF0413781, and Ministry of Industry and Information Technology of the People's Republic of China. The authors thank Yuhan Yang and Rui Huang from Tianjin University of Technology for their revision on the manuscript.

References

[1] M.R. Endsley, Toward a theory of situation awareness in dynamic systems, Hum. Factors 37 (1) (1995) 32–64, https://doi.org/10.1518/001872095779049543.
[2] Hooman Alavizadeh, et al., A survey on cyber situation-awareness systems: framework, techniques, and insights, ACM Comput. Surv. 55 (5) (2022) 1–37.
[3] Dongmei Zhao, Yaxing Wu, Hongbin Zhang, A situation awareness approach for network security using the fusion model, Mob. Inf. Syst. (2022) 2022.
[4] Mohammad Masum Billah, Jing Zhang, Tianchi Zhang, A method for vessel's trajectory prediction based on encoder decoder architecture, J. Mar. Sci. Eng. 10 (10) (2022) 1529.
[5] Jun An, et al., Transient stability margin prediction under the concept of security region of power systems based on the long short-term memory network and attention mechanism, Front. Energy Res. 10 (2022) 838791.
[6] L. Chen, M. Zheng, Z. Liu, M. Lv, L. Zhao, Z. Wang, SDAE+Bi-LSTM-based situation awareness algorithm for the CAN bus of intelligent connected vehicles, Electronics 11 (2022) 110.
[7] Boris N. Oreshkin, et al., N-BEATS: neural basis expansion analysis for interpretable time series forecasting, in: International Conference on Learning Representations, 2019.
[8] Xiaocai Zhang, Zhixun Zhao, Jinyan Li, ARDE-N-BEATS: an evolutionary deep learning framework for urban traffic flow prediction, IEEE Int. Things J. 10 (3) (2022) 2391–2403.
[9] Xiong Zhong, Xinsheng Zhang, Ping Zhang, Pipeline risk big data intelligent decision-making system based on machine learning and situation awareness, Neural Comput. Appl. 34 (18) (2022) 15221–15239.
[10] Syed M.H. Rizvi, Tahir Syed, Jalaluddin Qureshi, Real-time forecasting of petrol retail using dilated causal CNNs, J. Ambient Intell. Humaniz. Comput. (2022) 1–12.
[11] H. Wang, D. Zhao, X. Li, Research on network security situation assessment and forecasting technology, J. Web Eng. 19 (7–8) (2020) 1239–1266.
[12] Jie Ma, et al., Intent prediction of vessels in intersection waterway based on learning vessel motion patterns with early observations, Ocean Eng. 232 (2021) 109154.
[13] Ran Zhang, et al., A model of network security situation assessment based on BPNN optimized by SAA-SSA, Int. J. Digit. Crime Forensics (IJDCF) 14 (2) (2022) 1–18.
[14] Dongmei Zhao, Yaxing Wu, Hongbin Zhang, A situation awareness approach for network security using the fusion model, Mob. Inf. Syst. (2022) 2022.
[15] Jun An, et al., Transient stability margin prediction under the concept of security region of power systems based on the long short-term memory network and attention mechanism, Front. Energy Res. 10 (2022) 838791.
[16] Xiong Zhong, Xinsheng Zhang, Ping Zhang, Pipeline risk big data intelligent decision-making system based on machine learning and situation awareness, Neural Comput. Appl. 34 (18) (2022) 15221–15239.
[17] Li Jingzhao, Meng Yifan, Wang Jiwei, Multi-level safety situation awareness system for mines, Ind. Mine Autom. 46 (12) (2020) 1–6.
[18] Kun Yin, et al., Long-term prediction of network security situation through the use of the transformer-based model, IEEE Access 10 (2022) 56145–56157.
[19] Zhicheng Wen, et al., A network security situation awareness method based on GRU in big data environment, Int. J. Pattern Recognit. Artif. Intell. 37 (01) (2023) 2251018.
[20] Xuanzheng Wang, et al., EcoForecast: an interpretable data-driven approach for short-term macroeconomic forecasting using N-BEATS neural network, Eng. Appl. Artif. Intell. 114 (2022) 105072.
[21] Xiaobing Ma, et al., Decision-level machinery fault prognosis using N-BEATS-based degradation feature prediction and reconstruction, Mech. Syst. Signal Process. 198 (2023) 110435.
[22] Su Xu, et al., Solar cycle 25 prediction using N-BEATS, Astrophys. J. 947 (2) (2023) 50.
[23] M.T. Anwar, M.F. Islam, M.G.R. Alam, Forecasting meteorological solar irradiation using machine learning and N-BEATS architecture, in: Proceedings of the 2023 8th International Conference on Machine Learning Technologies, 2023, pp. 46–53.
[24] Boris N. Oreshkin, et al., N-BEATS neural network for mid-term electricity load forecasting, Appl. Energy 293 (2021) 116918.
[25] Abdul Khalique Shaikh, et al., Short term energy consumption forecasting using neural basis expansion analysis for interpretable time series, Sci. Rep. 12 (1) (2022) 22562.

[26] Ahmad El Majzoub, Fethi A. Rabhi, Walayat Hussain, Evaluating interpretable machine learning predictions for cryptocurrencies, Intell. Syst. Account. Finance Manag. (2023).
[27] H. Wen, J. Gu, J. Ma, L. Yuan, Z. Jin, Probabilistic load forecasting via neural basis expansion model based prediction intervals, IEEE Trans. Smart Grid 12 (4) (2021).
[28] Jatin Bedi, Yashwant Singh Patel, STOWP: a light-weight deep residual network integrated windowing strategy for storage workload prediction in cloud systems, Eng. Appl. Artif. Intell. 115 (2022) 105303.
[29] Biao Zhang, et al., Electricity price forecast based on the STL-TCN-NBEATS model, Heliyon 9 (1) (2023).
[30] J. Li, T. Lin, H. Du, Q. Li, X. Fu, X. Xu, A wind power prediction model based on optimized N-BEATS network with multivariate inputs, in: IEEE Power & Energy Society General Meeting (PESGM), IEEE, 2023, pp. 1–5.
[31] A. Karamchandani, A. Mozo, S. Vakaruk, S. Gómez-Canaval, J.E. Sierra-García, A. Pastor, Using N-BEATS ensembles to predict automated guided vehicle deviation, Appl. Intell. 53 (21) (2023) 26139–26204.
[32] Feihong Xu, Xianliang Teng, Jixiang Lu, Tao Zheng, Yulong Jin, Prediction of day-ahead electricity price based on N-BEATSx model optimized by SSA considering coupling between features, in: The Purple Mountain Forum on Smart Grid Protection and Control, Springer Nature Singapore, Singapore, 2022, pp. 178–194.
[33] Y. Song, G. Zhao, B. Zhang, H. Chen, W. Deng, W. Deng, An enhanced distributed differential evolution algorithm for portfolio optimization problems, Eng. Appl. Artif. Intell. 121 (2023) 106004.
[34] X. Zhou, X. Cai, H. Zhang, Z. Zhang, T. Jin, H. Chen, W. Deng, Multi-strategy competitive-cooperative co-evolutionary algorithm and its application, Inf. Sci. 635 (2023) 328–344.
[35] L.Y. Ma, J.M. Sun, NOx emission optimization based on SDAE prediction model and improved SSA, Proc. CSEE 42 (14) (2022) 5194–5202.
[36] Daniel R. MacNulty, L. David Mech, Douglas W. Smith, A proposed ethogram of large-carnivore predatory behavior, exemplified by the wolf, J. Mammal. 88 (3) (2007) 595–605.
[37] Hao Lin, Chundong Wang, Qingbo Hao, A novel personality detection method based on high-dimensional psycholinguistic features and improved distributed Gray Wolf Optimizer for feature selection, Inf. Process. Manag. 60 (2) (2023) 103217.
[38] K. Tadist, F. Mrabti, N.S. Nikolov, Z. Azeddine, N. Said, SDPSO: spark distributed PSO-based approach for feature selection and cancer disease prognosis, J. Big Data 8 (2021) 19.
[39] H. Chen, S. Tu, H. Xu, The application of improved grasshopper optimization algorithm to flight delay prediction-based on spark, in: Proceedings of the 15th International Conference on Complex, Intelligent and Software Intensive Systems (CISIS-2021), 2021, pp. 80–89.
[40] F.J. Solis, J.B. Wets, Minimization by random search techniques, Math. Oper. Res. 6 (1981) 19–30.
[41] M.J. Zhang, D.Y. Long, X. Wang, J. Yang, Research on convergence of grey wolf optimization algorithm based on Markov chain, Acta Electron. Sin. 48 (2020) 9–18.
[42] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1997) 67–82.
[43] Esmat Rashedi, Hossein Nezamabadi-pour, Saeid Saryazdi, GSA: a gravitational search algorithm, Inf. Sci. 179 (13) (2009) 2232–2248.
[44] S. Mirjalili, S.M. Mirjalili, A. Hatamlou, Multi-verse optimizer: a nature-inspired algorithm for global optimization, Neural Comput. Appl. 27 (2016) 495–513.
[45] S. Mirjalili, A.H. Gandomi, S.Z. Mirjalili, S. Saremi, H. Faris, S.M. Mirjalili, Salp swarm algorithm: a bio-inspired optimizer for engineering design problems, Adv. Eng. Softw. 114 (2017) 163–191.
[46] Seyedali Mirjalili, SCA: a sine cosine algorithm for solving optimization problems, Knowl.-Based Syst. 96 (15) (2016) 120–133.
[47] D. Simon, Biogeography-based optimization, IEEE Trans. Evol. Comput. 12 (6) (2008) 702–713.
[48] Bikash Das, V. Mukherjee, Debapriya Das, Student psychology based optimization algorithm: a new population based optimization algorithm for solving optimization problems, Adv. Eng. Softw. 146 (2020) 102804.
[49] K.G. Olivares, C. Challu, G. Marcjasz, R. Weron, A. Dubrawski, Neural basis expansion analysis with exogenous variables: forecasting electricity prices with NBEATSx, Int. J. Forecast. 39 (2) (2023) 884–900.