International Journal for Housing Science and Its Applications
Publish August 10, 2025. Volume 46, Issue 4 Pages 524-538

https://doi.org/10.70517/ijhsa46447

Deep learning-based load forecasting model for microgrids and its application in optimal dispatch strategy

Zhixiang Dai[1], Li Xu[1,*], Feng Wang[1], Mengjie Deng[2] and Taiwu Xia[1]
[1] Natural Gas Gathering and Transmission Engineering Technology Research Institute, PetroChina Southwest Oil & Gasfield Company, Chengdu, Sichuan, 610041, China
[2] Shunan Gas Mine, PetroChina Southwest Oil & Gasfield Company, Luzhou, Sichuan, 646099, China
Corresponding authors: (e-mail: snxuli@petrochina.com.cn).

Abstract Accurate load forecasting can not only help microgrids improve the utilization efficiency of energy resources, but also ensure the stability and reliability of power supply. In this paper, a deep learning-based load forecasting model for microgrids is proposed, and its application effect in different microgrids is verified through experiments. First, deep learning algorithms such as LSTM, BiGRU and CNN are used to construct a hybrid prediction model, and TVFEMD technique is introduced to signal decompose the load data to reduce the influence of noise. Through comparative experiments, the results show that on microgrid 1, the proposed model has a higher prediction accuracy with a minimum MAPE value of 2.0485% compared with the traditional methods, while on other microgrids, the model still maintains a more stable performance. In microgrid 3, the prediction results are more reliable in general, although there is a large error. Based on the experimental results of the model, this paper also discusses the interpretability of the model and its potential application in real microgrid scheduling. Ultimately, the proposed deep learning model can effectively improve the accuracy of microgrid load prediction with strong adaptability and stability.

Index Terms microgrid, load forecasting, deep learning, LSTM, CNN, TVFEMD

I. Introduction

In recent years, the power system has developed into a complex network system with centralized power generation and long-distance transmission [1]. With the increasing electricity load of the power system, the problems of high operating cost, difficult operation, and weak regulation ability of grid transmission are becoming more and more prominent [2], [3]. The large-scale power outages due to climatic reasons, such as the 2008 snowstorm blackout in southern China, the 2015 “rainbow” blackout, and the 2018 “mangosteen” blackout, reflect the vulnerability of the traditional large-scale power grid. In addition, the global energy crisis, environmental pollution and other issues have intensified, people gradually began to study the use of clean energy generation to make up for the shortcomings of the traditional energy-consuming power generation methods. Since then green power generation, efficient power transmission, flexible power microgrids have gradually entered the public's vision [4], [5].

Microgrid consists of a collection of various distributed power sources, energy storage units, loads, and monitoring and protection devices [6]. It has flexible operation and dispatchability, and can switch between two modes of grid-connected operation and islanded (autonomous) operation [7]. The increase in the number and penetration of distributed power sources in microgrids has a great impact on the voltage, network loss and short-circuit capacity of the distribution system, and data forecasting for microgrid power system optimization becomes extremely important due to the uncertainty of distributed power sources and the discontinuity of some wind and photovoltaic power generation [8], [9].

Power load forecasting is a key component of power system planning and operation, which involves accurate estimation of power demand over a future period of time [10]. This work is crucial for the reliability, economy and sustainability of the power system. Deep Learning Emerges-Deep learning techniques, especially Long Short-Term Memory Networks (LSTMs) and Convolutional Neural Networks (CNNs), have shown great potential in power load forecasting [11], [12]. These neural networks are capable of handling complex time series data and capturing patterns such as seasonality, weekdays and special events of loads [13]. Researchers have worked on developing uncertainty modeling and management methods to improve the reliability of forecasts. Smart microgrid management systems use advanced control algorithms and communication technologies to achieve real-time management of resources within the microgrid [14], [15]. The above research trends indicate that the field of power load forecasting and microgrid scheduling is full of innovations and opportunities [16]. With the emergence and development of new technologies, power systems will become more sustainable, intelligent and reliable.

In this paper, a hybrid model based on deep learning is proposed for microgrid load forecasting, which combines a convolutional neural network, a long and short-term memory network, and a bi-directional gated loop unit. Through the characteristics of deep learning, the temporal features in load data can be effectively captured to overcome the limitation that the traditional model cannot deal with nonlinear changes. In addition, the introduction of TVFEMD signal decomposition technique enables the load data to be better denoised and capture the detailed features in it. Through the combination of multiple optimization techniques, the model proposed in this paper is validated in multiple microgrid systems, which aims to improve the prediction accuracy and enhance the applicability of the model.

II. Deep learning-based load forecasting model for microgrids

II. A.Principles for forecasting microgrid loads

Practical experience shows that correct load forecasting is crucial to the operation and control of microgrids, which can effectively deploy energy resources and improve the reliability and economy of power supply. When predicting the load of microgrid, the following principles should be strictly observed.

II. A. 1) Data analysis

Real-time collection and analysis of microgrid daily load curve, weekly load curve, seasonal changes and other historical load data, through in-depth analysis of historical data, to discover the regularity and characteristics of the load, to provide a basis for future forecasting.

II. A. 2) Consideration of influencing factors

Load microgrid forecasting takes into account various factors that may affect the load, such as ambient temperature, holidays, and special events. All of the above factors will have an impact on load demand. Among them, hot weather will increase people's use of air conditioning, leading to an increase in load, and correlating the weather data with the load data can more accurately predict future load demand. Special circumstances such as sports events and large-scale conferences can lead to load fluctuations. Correcting or adjusting future loads based on historical data, relevant holiday and special event information can further enhance the reliability of forecasting results, and therefore need to be taken into account when building forecasting models.

II. A. 3) Multivariate models

In order to improve the accuracy of load forecasting, multivariate models can be used for forecasting, at this stage, it has been proved that the models that can be used for load forecasting are time series analysis, regression analysis, ANN, etc. The comprehensive use of a variety of models will help the staff to better capture the pattern of change of the load demand, so as to determine the focus of the subsequent work.

II. A. 4) Real-time adjustments

Considering the microgrid operation process, there may be equipment failure, weather changes and other unexpected factors, therefore, load forecasting is a dynamic process of constant change, only according to the actual situation, timely adjustment of the forecast results, in order to make the work give full play to its role, to ensure the stable operation of the microgrid.

II. A. 5) Refined management

To improve the accuracy of load forecasting, refined management methods can be used as appropriate, for example, zonal forecasting methods and load classification methods, which can be used to predict more accurately the load demand in each zone or type by dividing the microgrid into different zones or types, leading to better energy deployment and management.

II. B.Methods for forecasting microgrid loads

Deep learning is an ANN-based machine learning method characterized by the ability to learn high-level feature representations of data through multi-layer neural networks. In network load forecasting, Recurrent Neural Network (RNN), Long Short-Term Memory Network (LSTM) deep learning models can all quickly and accurately capture the temporal features of the load data to improve the accuracy and stability of the forecast. The specific steps for predicting microgrid load using this technique are as follows:

In the first step, collect microgrid historical load data, including but not limited to load time series data, weather data, holiday data, the collected data will be used as model input features to help the model learn the changing law of load [17].

In the second step, the data are cleaned and preprocessed to deal with missing values, outliers and do smoothing on the data to ensure the quality of the data.

In the third step, after preprocessing, extract and construct features suitable for deep learning model, if there is no special requirement, give priority to the use of time features, weather features, holiday features, and subsequently, do standardization or normalization operation on the extracted features.

In the fourth step, according to the actual situation of microgrids, load forecasting requirements, and data size, scientifically select the deep learning models used for load forecasting, such as RNN, LSTM, and all the above models are able to capture the long-term dependence and complex relationships of time series data.

In the fifth step, the dataset is divided into a training set and a test set, where the training set is responsible for model training and the test set is used to verify the performance of the model. During the training process, the established hyperparameters should be adjusted according to the model performance to improve the accuracy and generalization ability of the model.

In the sixth step, the Mean Absolute Percentage Error (MAPE) metrics are comprehensively used to assess the model performance and understand the prediction accuracy and stability of the model.

In the seventh step, the trained model is deployed to the actual microgrid system, and the prediction of the load is completed using the period, which is combined with the real-time data to realize the accurate prediction and scheduling of the microgrid load. It should be noted that deep learning models are usually black-box models, so when selecting and using the model, the explanatory and interpretable nature of the model is required to ensure that the prediction results are in line with the actual situation.

II. C.TVFEMD-MFSMA-CNN-BiGRU-LSTM load forecasting model

II. C. 1) Convolutional Neural Networks

Convolutional neural network is a type of feed forward neural network which is a type of deep learning. CNN inspires the inspiration of Receptive Field (RF) in Neurology. The pixels on the map output from each layer of the corresponding CNN will be convolved to the next layer and the Receptive Field will get bigger and bigger as the convolution kernel increases. The formula for the RF is:

$$RF_i = (RF_{i+1} - 1) \cdot stride_i + K_{size_i} \tag{1}$$

where RF_i denotes the size of the sensory field in the i th layer, $stride_i$ is the step size of the convolution, and K_{sizei} is the size of the convolution kernel in the i th layer.

II. C. 2) LSTM

Long and short-term memory neural network (LSTM) simulates the human brain to learn data effectively.LSTM is improved on the basis of recurrent neural network (RNN), and its specific structure consists of forgetting gate, input gate and output gate. The forgetting gate can selectively forget the historical information in the sequence, the input gate can fuse the current moment information with the historical information, and the output gate can decide the output state of the hidden layer, which overcomes the problem of gradient disappearance of RNN in dealing with long time sequences, and strengthens the ability to deal with long time sequences.

The formula for the forgetting gate is shown in (2):

$$F_t = \sigma(W_F \cdot [h_{t-1}, x_t] + b_F) \tag{2}$$

The equations for the input gate are shown in (3) and (4):

$$A_t = \sigma(W_A \cdot [h_{t-1}, x_t] + b_A) \tag{3}$$

$$B_t = \sigma(W_B \cdot [h_{t-1}, x_t] + b_B) \tag{4}$$

The formulas for the output gates are shown in (5) and (6):

$$O_t = \sigma(W_0 \cdot [h_{t-1}, x_t] + b_0) \tag{5}$$

$$h_t = O_t \times \tanh(C_t) \tag{6}$$

where, W_F, W_A, W_B, W_O denote the weights, b_F, b_A, b_B, b_O denote the bias, h_{t-1} denote the outputs of the basic unit of the LSTM in the previous moment, x_i denotes the input (load feature) at the current moment, and C_i denotes the updated cell state. F_t is the output of the forgetting gate, which is used to discard information, the product of A_t and B_t is the output of the input gate, and h_t is the output value of the basic unit of the LSTM at moment t, and [] inside the formula denotes the vector splicing, σ denotes the sigmoid function, and tanh denotes the hyperbolic tangent function, and the formulas are shown in Eqs. (7) and (8), respectively:

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{7}$$

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{8}$$

By using LSTM to establish a short-term power load forecasting model and verified through simulation, compared with BP neural network, LSTM has a higher prediction accuracy, however, although LSTM solves the gradient disappearance problem existing in RNN, the calculation is more complex, and the algorithm consumes a longer time.

II. C. 3) BiGRU

Gated neural network (GRU) is a variant of LSTM, which consists of a reset gate and an update gate. GRU fuses the input gate and forgetting gate of LSTM coupled into an update gate, whose role is to control whether to retain the state of the previous moment and how much information is retained, and the role of the reset gate is to combine the current state with the previous information. The network forward propagation formula is shown in the following equation:

$$r_t = \sigma(U_r h_{t-1} + W_r x_t) \tag{9}$$

$$z_t = \sigma(U_z h_{t-1} + W_z x_t) \tag{10}$$

$$h_t = \tanh(W x_t + U(r_t \square h_{t-1)})) \tag{11}$$

$$h_t = (1 - z_t) \square h_{t-1} + z_t \square h_t \tag{12}$$

where r_i is the reset gate, z_t is the update gate, h_t is the vector of candidate hidden layers, h_t is the output of the hidden layer, W_r, U_r is the weight of the reset gate, W_z, U_z are the update gate's weights, W, U are the weights of candidate hidden states, tanh is the hyperbolic tangent function, $\square$ is the Hadamard product of two matrices, and σ is the sigmoid function.

By using cuckoo search algorithm (CS) to optimize the parameters of GRU, a higher accuracy of short-term power load prediction is obtained, and GRU simplifies the structure of LSTM, which improves the computational efficiency and enhances the information processing capability while maintaining the similar prediction effect of LSTM.

II. C. 4) CNN-BiGRU-LSTM hybrid network prediction models

In this paper, LSTM, CNN-BiGRU are combined to form a hybrid prediction model, which is mainly composed of input layer, CNN layer, BiGRU layer, LSTM layer and output layer. The specific model is shown in Figure 1.

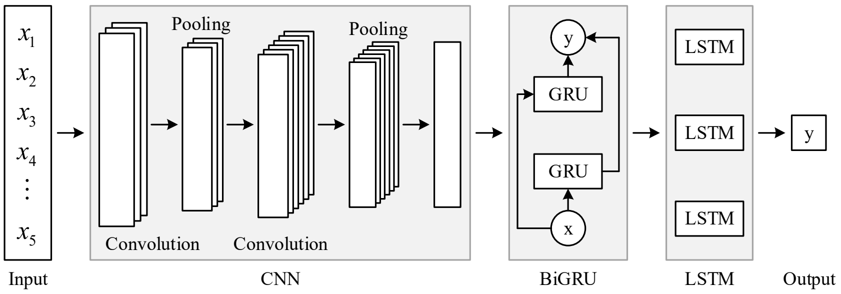


Figure 1: Prediction model of CNN-BiGRU-LSTM Hybrid Network

II. D.TVFEMD-based signal decomposition

TVFEMD makes up for the shortcomings of the previous two by combining the techniques of variable filtering and EMD.TVFEMD can adapt to the time-varying characteristics of signals by dynamically adjusting the parameters of the filters to reduce the sensitivity to noise and sudden changes, thus improving the stability and accuracy of the decomposition results. In addition, TVFEMD can take into account the changing characteristics of the signal in different time periods, which has better performance in dealing with non-stationary signals, and also provides flexible parameter adjustment to adapt to different types of signals and application scenarios [18]. The implementation of TVFEMD is described below.

(1) Firstly, the Hilbert transform (HT) is proposed, which is a mathematical transformation converts a signal in the real domain into a signal in the complex domain, which contains the amplitude, phase and other information of the original signal. For a real signal $x(t)$, its HT transform can be expressed as:

$$H(x(t)) = 1/\pi \int_{-\infty}^{+\infty} \frac{x(\tau)}{t-\tau} d\tau \tag{13}$$

(2) Find the instantaneous amplitude $A_{inst}(t)$ and the instantaneous phase $\varphi_{inst}(t)$:

$$A_{inst}(t) = \sqrt{x(t)^2 + H(x(t))^2} \tag{14}$$

$$\varphi_{inst}(t) = \tan^{-1}\left[\frac{x(t)}{H(x(t))}\right] \tag{15}$$

(3) Find the maximum, and minimum values of the instantaneous amplitude, and derive the two curves $l_1(t)$ and $l_2(t)$ from interpolation estimation, and compute $\theta_1(t)$ and $\theta_2(t)$ from the following equations:

$$\theta_1 = \frac{l_1(t) + l_2(t)}{2} \tag{16}$$

$$\theta_2 = \frac{l_1(t) - l_2(t)}{2} \tag{17}$$

(4) Interpolating $A_{inst}(t_{\max})^2 \mu'(t_{\max})$ and $A_{inst}(t_{\min})^2 \mu'(t_{\min})$ to get $\delta_1(t)$ and $\delta_2(t)$, and then proceed to compute $\mu_1(t)$ and $\mu_2'(t)$:

$$\mu_1'(t) = \frac{\delta_1(t)}{2\theta_1^2(t) - 2\theta_1(t)\theta_2(t)} + \frac{\delta_2(t)}{2\theta_1^2(t) + 2\theta_1(t)\theta_2(t)} \tag{18}$$

$$\mu_2'(t) = \frac{\delta_1(t)}{2\theta_2^2(t) - 2\theta_1(t)\theta_2(t)} + \frac{\delta_2(t)}{2\theta_2^2(t) + 2\theta_1(t)\theta_2(t)} \tag{19}$$

(5) Calculate the local cutoff frequency:

$$\mu_{bis}'(t) = \frac{\mu_1'(t) + \mu_2'(t)}{2} \tag{20}$$

(6) Use a B-spline approximation filter for $x(t)$, denoted $x'(t)$, and compute the signal $s(t)$:

$$s(t) = \cos\left[\int \mu_{bis}'(t)dt\right] \tag{21}$$

(7) End the loop when the end condition is satisfied, otherwise loop through steps (1)-(6). The judgment criteria are noted as:

$$\rho(t) = \frac{B_{Loug}(t)}{\overline{\mu_{average}}} \tag{22}$$

where $B_{Loug}(t)$ denotes the instantaneous bandwidth, which represents the spectral characteristics of the signal at that point in time, and $\overline{\mu_{average}}$ denotes the weighted average of the instantaneous frequencies, which can be defined according to the specific needs and applications.

II. E. MFSMA Optimization Tuning

II. E. 1) Sticky Bacteria Algorithm

The slime mold algorithm is a new swarm intelligence algorithm proposed in 2020. It is inspired by the slime molds in the biological world. Slime molds are a class of eukaryotic organisms with unique intelligent behaviors and adaptations that enable them to find food, build effective channel networks and make survival decisions in complex environments.

Myxomycetes are also able to dynamically adjust their search patterns according to the quality of the food source. When the quality of the food source is high, the slime mold will adopt an area-limited search method, focusing on the discovered food sources. If the density of the initially discovered food source is low, the slime mold will leave that food source and search for other alternative food sources in the area. This adaptive search strategy is particularly evident when food blocks of varying quality are present in an area. A conceptual diagram of a slime mold as it acquires food is shown in Figure 2.

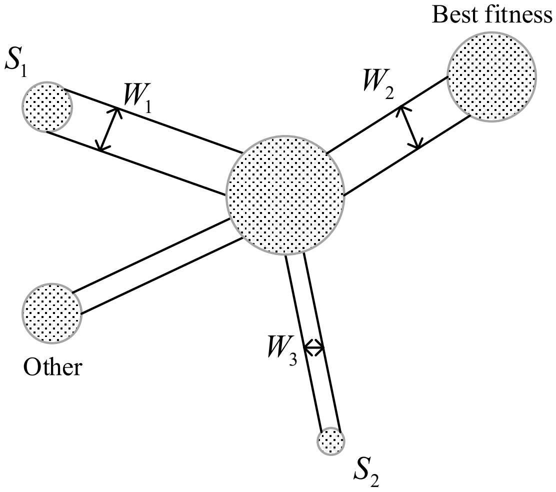


Figure 2: Foraging strategies of Slime molds

where the green dots represent the locally optimal food source and W is the width of the venous tube, it can be seen that the higher the quality of the food source, the wider the venous tube will be. Taking advantage of this property, the slime mold algorithm can be utilized to find the extreme value of the solution function. The unique properties of the slime mold community not only can fully utilize the solution space, but also can find the extreme value quickly. Therefore, this paper focuses on the slime mold algorithm. Next, mathematical modeling of the slime mold system will be carried out:

$$X(t+1)=\begin{cases}X_b(t)+v_b\cdot(W\cdot X_A(t)-X_B(t)), r<p\\ v_c\cdot X(t), r\geq p\end{cases} \tag{23}$$

Equation (23) is the main search formula of the slime mold algorithm. Where t represents the current number of iterations, m represents the number of slime molds, $X_{b(t)}$ represents the optimal solution under the current number of iterations, $X_{A(t)}$ and $X_B(t)$ are the two stochastic solutions under the current number of iterations, and $X(t)$ denotes the current solution. $r\in rand[0,1]$. The W, v_b, v_c, p are the four important parameters. The four formulas are explained below:

$$p=tanh\,|\,S(i)-DF\,| \tag{24}$$

$$a=tanh^{-1}(-(\frac{t}{T}+1)), v_b\in[-a,a] \tag{25}$$

$$W(SI(i)) = \begin{cases} 1 + r \cdot log(\frac{bF - S(i)}{bF - wf} + 1), & i < \frac{m}{2} \\ 1 - r \cdot log(\frac{bF - s(i)}{bF - wf} + 1), & i \geq \frac{m}{2} \end{cases} \tag{26}$$

$$SI(i) = sort(S) \tag{27}$$

where v_c is a coefficient that oscillates between [-1, 1] and eventually converges to 0 as the number of iterations increases. W denotes the width of the venous pipeline, $S(i)$ denotes the fitness value of the current individual solution, T denotes the maximum number of iterations, and DF, bF, wf denotes the current optimal fitness value throughout the entire iteration process, the optimal fitness value at the current number of iterations, and the worst fitness value in the current iteration process, respectively. $SI(i)$ denotes the ordering of the fitness values of the slime mold population after one iteration. This part mainly models the thickening or thinning of the veins of the slime molds depending on the food concentration. However, the slime molds do not always move towards the food source with high concentration, and some individuals in the slime mold community are always separated to search for other food sources, partly to avoid falling into the local optimal solution and partly to increase the search space. To summarize, the position update formula for a population of slime molds is as follows:

$$X(t+1) = \begin{cases} X_b(t) + v_b \cdot (W \cdot X_A(t) - X_B(t)), r < p \\ v_c \cdot X(t), r \geq p \\ rand \cdot (UB - LB) + LB, rand < z \end{cases} \tag{28}$$

where UB, LB denotes the upper and lower bounds of the solution space, and once the random number is small z, the individual searches for the food source at a random location. The unique search mechanism of the Sticky Mushroom algorithm ensures both a faster convergence rate and strong robustness, which is a clear advantage among the emerging swarm intelligence algorithms.

II. E. 2) MFSMA-based tuning methods

In this paper, we use MFSMA to tune the deep hybrid model CNN-BiGRU-LSTM. According to the introduction of CNN-BiGRU-LSTM in previous sections, it is known that this hybrid prediction model contains several hyperparameters. They include learning rate, batch size, number of training times, number of filters, number of BiGRU hidden layer neurons and number of LSTM hidden layer neurons. So in this paper, the data of these six dimensions are given to MFSMA for optimization, so as to achieve the effect of tuning parameters. The specific flowchart of CNN-BiGRU-LSTM tuning using MFSMA is shown in Figure 3.

The specific MFSMA optimization parameter flow is described below:

Step1: Initialize the algorithm parameters. Determine the population size, the number of iterations, the hyperparameters, and the upper and lower bounds of the search range.

Step2: Initialize each individual in the slime mold population. Each individual $X_i = (l, s, t, n, h_1, h_2)$, records such six-dimensional data. Where l represents the learning rate of the hybrid prediction model, s represents the batch size, t represents the number of times the model has been trained, n represents the number of filters, and h_1 and h_2 represent the number of neurons in the BiGRU and LSTM layers. The upper and lower bounds in this paper are: learning rate [le-3, le-1], batch size [16, 256], number of trainings [10, 100] number of filters [16, 128], number of BiGRU neurons [1, 100], and number of LSTM neurons [1, 100].

Step3: Determine the fitness value function of MFSMA. Train the hybrid prediction model using the hyperparameters obtained from the initialization in Step2. Divide the data into a training set and a validation set. Input the training set into the prediction model for training, and after reaching the number of iterations, output the training sample prediction value and the validation sample prediction value, then the fitness value of X_i can be expressed as:

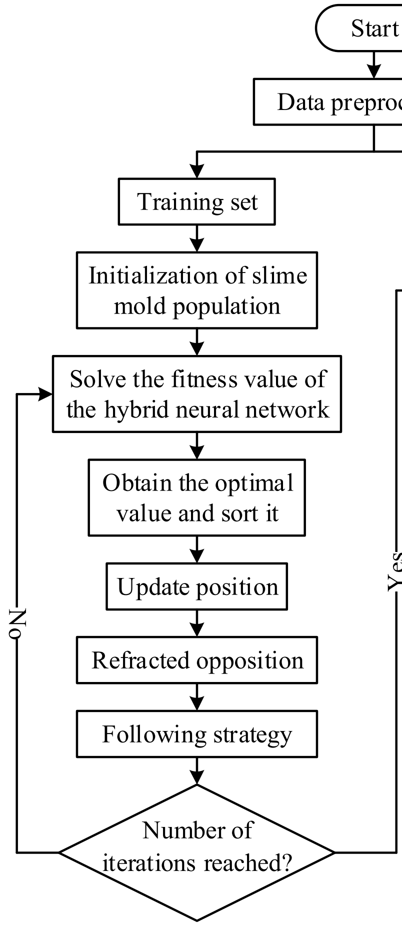

Start
Data preprocessing
Training set
Test set
Initialization of slime mold population
Hybrid prediction model
Solve the fitness value of the hybrid neural network
Prediction result
End
Obtain the optimal value and sort it
Update position
Refracted opposition
Following strategy
Number of iterations reached?
No
Yes

Figure 3: Flowchart of the MFMSA optimized hybrid prediction model

$$F = \frac{1}{2} \cdot \left(\sum_{j=1}^{J} \frac{\hat{y}_t^j - y_t^j}{y_t^j} + \sum_{k=1}^{K} \frac{\hat{y}_v^k - y_v^k}{y_v^k} \right) \tag{29}$$

where $\hat{y}_t^j$ is the training sample output value, $\hat{y}_v^k$ is the validation sample output value, and y_t^j is the training expected output value y_v^k is the validation expected output value. That is, the fitness value includes both the error during the training sample and the error of the validation sample, and both of them have the same weight, so that the model capability can be fully quantified into the fitness value, which is conducive to the optimization of the slime mold population.

Step4: Calculate the fitness value, and then rank the individual advantages and disadvantages according to the fitness value, and determine the current optimal individual and the global optimal solution.

Step5: Update the position using three steps. The first step is the original position update formula of the slime mold algorithm. The second step is the refractive opposition position update. The third part is the Bottle Sea Sheath following strategy position update. After that, Step4 is repeated.

Step6: After satisfying the iteration number requirement of MFSMA, the optimal result is output and set into the CNN-BiGRU-LSTM hybrid negative measurement model, then the model is trained with data and finally the load prediction value is output.

II. F. Analysis of microgrid load forecasting results

II. F. 1) Error evaluation indicators

The difference between the load prediction value and the actual value is called the prediction error, the lower the prediction error, the better, the lower the error, then the model is more effective [19]. Let y_i represent the actual load value and $\hat{y}_i$ represent the predicted load value, there are various methods of prediction error, and in this paper, we mainly utilize the mean absolute percentage error (MAPE):

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|y_i - \hat{y}_i\right| \tag{30}$$

II. F. 2) Comparison of prediction errors

For these four microgrids, on microgrid 1, this paper's model gives better predictions than GBDT and deep residual networks, with the smallest error and the best fit. On microgrid 2 and microgrid 3, this paper's model is less effective and does not give the best results. The results of this paper's model on microgrid 4 are also not the best. This is because microgrid prediction itself is more challenging.

Table 1 shows the microgrid prediction error MAPE values, the table gives the specific load prediction error values for these four microgrids on holidays and all weekend breaks, it can also be seen from the table that on microgrid 1, this paper's model obtains the smallest error, with MAPEs of 2.0485%, 16.7859%, 13.8485%, 9.3855% for the seven sample days, 11.5485%, 1.4156%, and 1.0485%. On microgrid 3, the prediction errors of this paper's model are particularly large on August 18 and August 19, which are 28.3485 and 23.3485, respectively. While for the other two microgrids, the prediction effect of this paper's model is not the worst, and it can be said that the prediction error is relatively stable, with small fluctuations in the error, and no particularly large error.

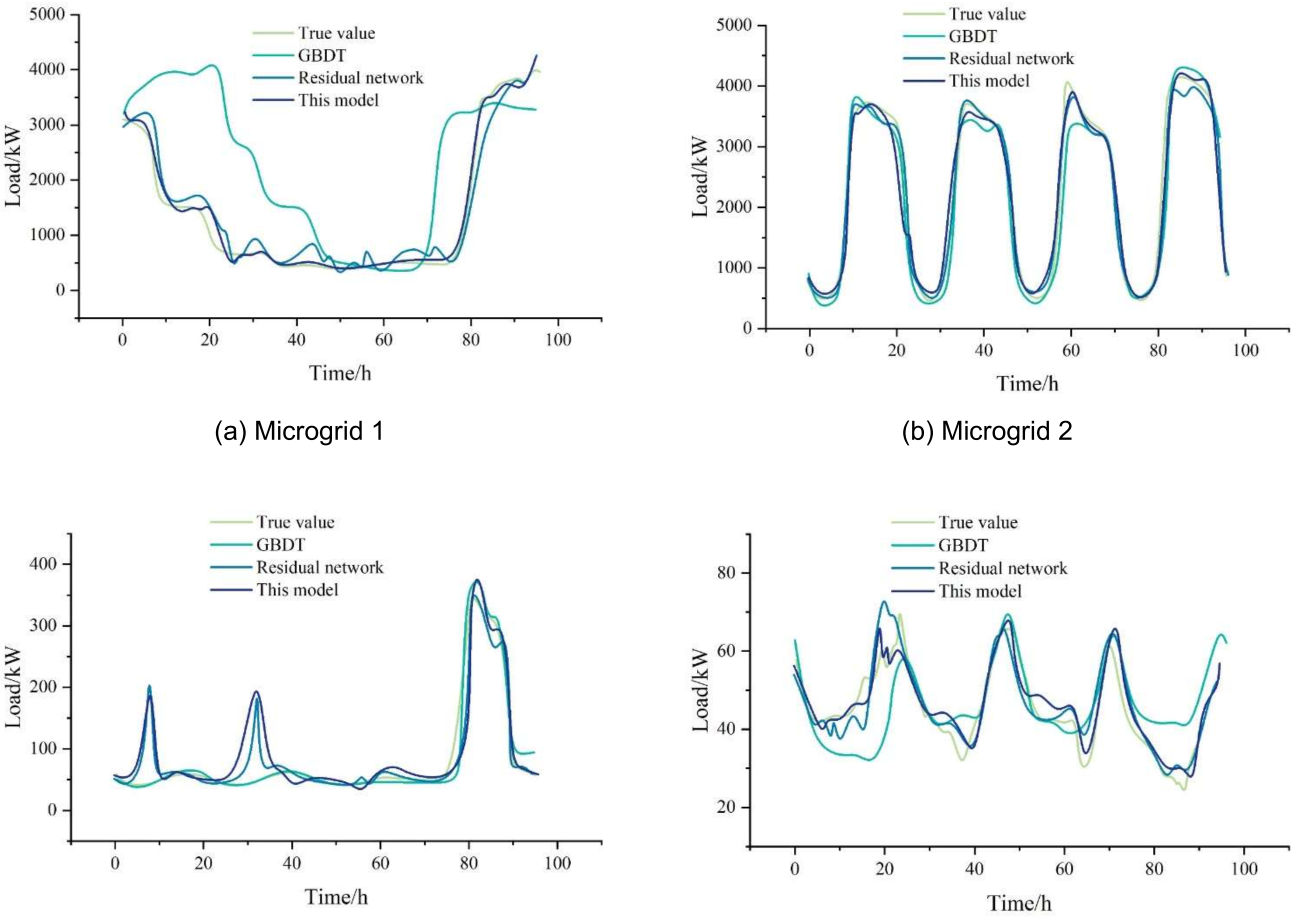
Table 1: Microgrid prediction error MAPE value

Date		Sunday	Statutory holidays (including Saturday)			Sunday (Work day)	Saturday	Sunday
		8-14	8-18	8-19	8-20	8-21	8-25	8-26
Microgrid 1	GBDT	1.9485	174.0485	209.7498	9.4855	164.9485	7.3452	1.6458
	Deep residual network	2.2369	28.6486	38.6152	37.0485	12.5136	2.1485	2.0248
	This model	2.0485	16.7859	13.8485	9.3855	11.5485	1.4156	1.0485
Microgrid 2	GBDT	4.8486	6.4048	5.9485	6.5248	7.7399	28.7458	3.8698
	Deep residual network	5.8665	8.3489	7.7388	5.6325	7.7493	9.1642	12.1485
	This model	5.5169	11.4855	6.5482	4.3115	8.1548	7.0385	8.2645
Microgrid 3	GBDT	10.1658	18.6485	9.2015	11.4485	15.1265	7.5152	5.6948
	Deep residual network	7.9585	34.2699	22.4985	10.1685	11.8425	8.6185	18.9645
	This model	6.1854	**28.3485**	**23.3485**	9.9758	10.2486	7.9186	3.6152
Microgrid 4	GBDT	22.4685	24.5169	10.8485	9.2348	32.5185	18.2869	12.0458
	Deep residual network	13.1785	10.8596	9.8482	8.1548	10.3645	10.6849	7.3484
	This model	14.8486	9.5469	9.9345	9.6385	8.4686	7.3448	6.3469

II. F. 3) Load Forecast Curves

The good performance of the neural network model proposed in this paper can be demonstrated by the prediction of legal holidays or Saturdays and Sundays, as well as peak and valley loads. Fig. 4 shows the load prediction curves for legal holidays of microgrids, and Figs. (a)-(d) represent microgrids 1-4, respectively.Load prediction during legal holidays of four microgrids is given for the period from 2023-8-18 to 2023-8-20.

For microgrid 1, by observing the fitting effect of actual and forecast values, it can be seen that the load curve of this paper's model fits significantly better than that of GBDT, with less fluctuation than that of the load curve of the deep residuals, and is more in line with the trend of the load change of the actual values, and the mean value of the error is 156.745. For microgrid 2, by observing the fitting effect of actual and forecast values, it can be seen that the load curve of this paper's model's load profile fits better than GBDT and depth residuals at the peak, has a lower prediction error with a mean error value of 11.979, but the prediction errors at other time points are larger. For microgrid 4, the model in this paper did not perform well in predicting holidays, but instead caused a large error, and the prediction results for weekends were not satisfactory. For microgrid 8, it can be seen that the fluctuation of this load curve itself is very obvious, so the prediction results of this paper's model also have more greatly fluctuation.



(a) Microgrid 1

(b) Microgrid 2

(c) Microgrid 3

(d) Microgrid 4

Figure 4: Forecast of statutory holiday load of microgrid

III. SSA-based optimal scheduling strategy for microgrid energy storage

III. A. Optimized scheduling model for microgrid systems

Multi-microgrid system is an extension of single microgrid. Therefore, this chapter constructs an optimal scheduling model for multi-microgrid systems based on the energy storage scheduling of microgrids and considering the power interaction between microgrids, so as to improve the utilization efficiency of renewable energy and reduce the interaction between microgrids and the main grid.

For the optimal scheduling problem of multi-microgrid system, it can be constructed into a multi-objective optimization model. In this chapter, based on the load forecasting results of each microgrid within the system, the renewable energy output of each microgrid, inter-microgrid power interaction and other information are taken into account to minimize the cost of each microgrid's own operation within the system. Since there is a competitive relationship between each microgrid for energy interaction, the inter-microgrid interaction cost and losses are added to the operating cost as shown in equation (31) as the operating cost of microgrid k per unit dispatch interval:

$$\begin{aligned}\min Cost_{k,MG} &= Cost_{k,DG} + Cost_{k,ESS} + Cost_{k,loss} \\ &+ Cost_{k,grid} - \sum_{n=1,n\neq k}^{K} Cost_{kn}\end{aligned} \tag{31}$$

$$Cost_{k,DG} = \sum_{i=1}^{G}(Cost_{k,DG}^{i.OM} + Cost_{k,DG}^{i.DEP}) \tag{32}$$

$$Cost_{k,ESS} = \gamma_{k,ESS} \times |P_{k,ESS}| \tag{33}$$

$$Cost_{kJoss} = \delta_{loss} \sum_{l=1}^{s} P_{k,loss}^{l} \tag{34}$$

$$Cost_{k.grid} = \begin{cases} R_{gridsell} \times P_{k.grid}, & P_{k.grid} \geq 0 \\ R_{gridbuy} \times P_{k.grid}, & P_{k,grid} < 0 \end{cases} \tag{35}$$

$$Cost_{kn} = -Cost_{nk} = \theta_{kn} \times P_{kn} \tag{36}$$

where in, $Cost_{k,DG}$ is the operating cost of the distributed generation unit in the microgrid k, the expression is shown in (32), where G is the number of distributed generators in the microgrid, and the $Cost_{k.DG}^{i.OM}$ and $Cost_{k,DG}^{i.DEP}$ are the maintenance cost and depreciation cost of the first i distributed power generation unit in the microgrid, respectively, and the calculation method is consistent with the calculation method in the microgrid model. Eqs. (33) and (34) are the formulas for the loss cost $Cost_{k,ESS}$ and the network loss cost $Cost_{kJoss}$ of the energy storage system in the microgrid, respectively. Eq. (35) and Eq. (36) are the calculation formulas of the interaction cost $cost_{k,grid}$ with the main power grid, the cost of energy interaction between the microgrid k and the microgrid n $cost_{kn}$, respectively, where $P_{k,gind}$ is the interaction energy between the multi-microgrid system and the main power grid, and the main grid supplies power to the system when its value is greater than 0. θ_{kn} and P_{kn} denote the trading tariff and the interacting power between microgrids and microgrids, respectively.

In order to guarantee the efficiency and stability of the multi-microgrid system, certain constraints should be satisfied. Like the single microgrid model, the power balance constraints of each microgrid, the protection control constraints of the energy storage system, and the output constraints of each micro-source are specified. Therefore, the constraints of the multi-microgrid system consider the internal interaction power and add the power balance constraints and the inter-microgrid interaction power constraints as follows:

(1) Power balance constraint: the distributed power output and interacting power in each microgrid within the system should meet the load side demand:

$$\sum_{i=1}^{G} P_{k,DG}^{i} + P_{k,ESS} + P_{k,grid} + \sum_{n=1,nxk}^{K} P_{kn} = P_{k,kotd} + P_{k,loss} \tag{37}$$

(2) Upper and lower bound constraints on the interaction power: the power flowing between the microgrids is between the maximum value of the tradable power between the two parties, $P_{kn}^{\max}$, and the minimum value, $P_{kn}^{\min}$:

$$P_{kn}^{\min} \leq P_{kn} \leq P_{kn}^{\max} \tag{38}$$

III. B. Algorithm for solving the microgrid energy storage scheduling model

III. B. 1) Pareto-based multi-objective sparrow search algorithm

Multi-objective problems need to optimize multiple objective problems, and usually there is no single solution that can optimize all the objectives at the same time. Therefore, in this paper, the Pareto dominance relation is used to obtain the Pareto optimal frontier consisting of non-inferior solutions, and the global optimal solution is selected through the Pareto frontier. The mathematical model for a multi-objective optimization problem with z-dimensional objective function and dim-dimensional variables is shown in Eq. (39):

$$\begin{gathered} \min F(x) = \{ f_1(x), f_2(x), \cdots, f_z(x) \} \\ s.t. \begin{cases} h_i(x) \leq 0, i = 1,2,3,\ldots,n \\ g_j(x) = 0, j = 1,2,3,\ldots,m \\ x \in [X_{\min}, X_{\max}] \end{cases} \end{gathered} \tag{39}$$

where x is the decision variable in dimension, $F(x)$ is the objective function in dimension z, and $x_{\min}, x_{\max}$ are the upper and lower bounds on the values of the decision variable, respectively. For this mathematical model, the relevant definition of Pareto is as follows:

Pareto dictates that for $\forall p \in [1,z]$, there is $f_p(x_1) \leq f_p(x_2)$ and $\exists q \in [1,z]$ such that $f_q(x_1) < f_q(x_2)$, then x_1 Pareto dominates x_2, denoted as $x_1 \prec x_2$.

Pareto optimal: if there is no vector x dominating the vector x^* in the decision space, it is called a Pareto optimal solution.

Pareto front: the surface formed by the values of the corresponding objective function is a Pareto front.

III. B. 2) Adaptive Grid-Based Multi-Objective Sparrow Search Algorithm

Since there is no direct priority relationship between the individuals in the external archive set, it is difficult to determine the optimal solution for the sparrow population. Therefore, the adaptive grid algorithm is used in the sparrow search algorithm to partition the dominant solutions in the external archive set into multiple grids, take the number of populations contained in each grid as the density information of the individuals, and then select the individuals in the sparse space as the global optimal solution according to the density information, thus ensuring the diversity of the dominant solutions and improving the algorithm's global search capability.

The non-dominated solutions in the external archive set are updated with the number of iterations, so the algorithm adaptively adjusts the grid size to update the positioning of individuals in the archive set to ensure that the population number of the grid where each individual is located is greater than 1. The three-dimensional objective space minimization optimization problem is used as an example to illustrate the process of generating the grid information by the adaptive grid algorithm:

Step 1: Calculate the boundary values of the fitness functions f_1, f_2, f_3 at the t th iteration: $f_1 \in (\min f_1^{'}, \max f_1^{'}), f_2 \in (\min f_2^{'}, \max f_2^{'})$ and $f_3 \in (\min f_3^{'}, \max f_3^{'})$.

Step 2: Equalize the target space into $Num(Num = D \times D \times D)$ lattices and compute the modes of the lattices respectively:

$$\Delta f_1^{'} = \frac{\max f_1^{'} - \min f_1^{'}}{D} \tag{40}$$

$$\Delta f_2^{'} = \frac{\max f_2^{'} - \min f_2^{'}}{D} \tag{41}$$

$$\Delta f_3^{'} = \frac{\max f_3^{'} - \min f_3^{'}}{D} \tag{42}$$

Step 3: Iterate over the populations in the external archive set and number them, the 3D sequence pairs of individual i are shown in equation (43):

$$\left(Int\left(\frac{f_1^i - \min f_1^{'}}{\Delta f_1^{'}}\right)+1 \quad Int\left(\frac{f_2^i - \min f_2^{'}}{\Delta f_2^{'}}\right)+1 \quad Int\left(\frac{f_3^i - \min f_3^{'}}{\Delta f_3^{'}}\right)+1 \right) \tag{43}$$

where $Int(\cdot)$ is the rounding function.

Step 4: Calculate the grid information and the density information of individuals.

In order to make the sparrow optimal individuals fully utilize the non-dominated solution information in the external archive set, for the individuals in the archive set, if the density information is lower, the probability of being selected is higher, and vice versa, the probability is lower. This can improve the diversity of nondominated solutions and avoid falling into local optimal solutions during the optimization search process.

III. B. 3) External archive set maintenance strategy

As the number of iterations of the algorithm increases, more and more nondominated solutions are saved in the external archive set, so the size of the external archive set needs to be set. In order to maintain the diversity of solutions in the archive set, this paper adopts a truncation algorithm based on the adaptive grid algorithm to eliminate redundant individuals and ensure that the number of non-inferior solutions in the archive set does not exceed the upper limit. The size of the external archive set is set to S_c , and when the number of particles in the archive set $|A_{t+1}| > S_c$, the lattice with a population greater than 1 is deleted from the redundant individuals according to Eq. (44):

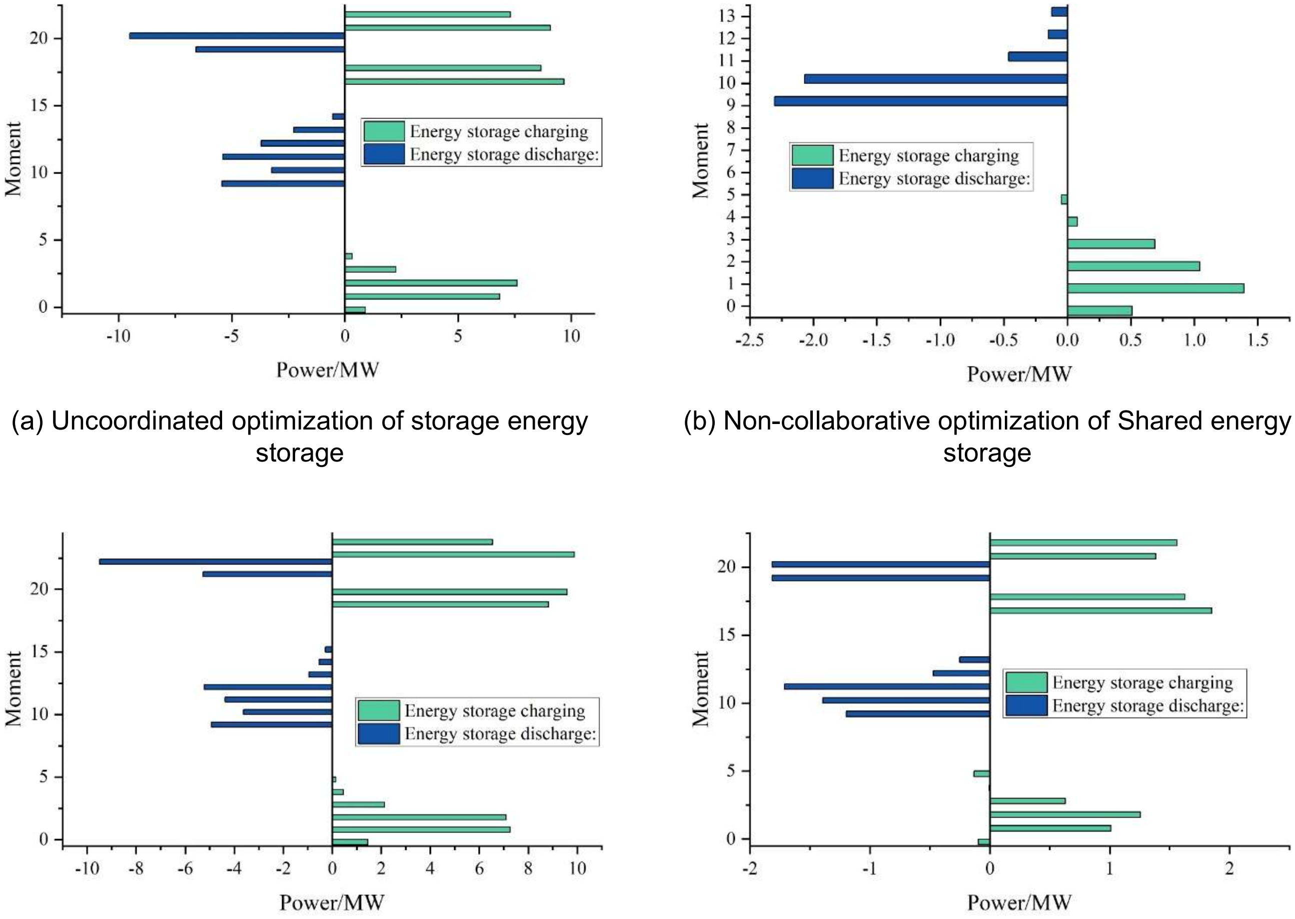
$$P_N = Int\left(\frac{|A_{t+1}| - S_c}{|A_{t+1}|} \times Grid[j] + 0.5\right) \tag{44}$$

where $Grid[j]$ is the number of population individuals contained in grid j and P_N is the number of individuals to be deleted from this grid.

III. C. Analysis of the results of the optimized energy storage scheduling methods

III. C. 1) Energy storage charge/discharge curves

The optimization effect before and after the implementation of the energy storage optimization scheduling method is shown in Fig. 5, Figs. (a) and (b) are non-co-optimization scheduling strategies, and Figs. (c) and (d) are co-optimization strategies. Under the non-co-optimization mode, the shared energy storage and plant storage carry out decision-making independently according to their own objectives, and the charging and discharging behaviors of the shared energy storage follow the change of the electricity price, charging at the moment of low electricity price (00:00-08:00 hours), and discharging arbitrage in the peak electricity price interval (09:00-14:00 hours), when the maximum power of discharging is close to 2.5MW. The energy storage of the plant and station follows the load change, and the load valley is charged to absorb wind power, such as (00:00-08:00, 16:00-17:00 and 21:00-22:00), and the load is increased to discharge to reduce the output of thermal power and reduce costs, such as (09:00-15:00 and 20:00-21:00). It can be seen that in the non-cooperative mode, the charging and discharging selection of bilateral energy storage is completely independent of its own goals.



(a) Uncoordinated optimization of storage energy storage

(b) Non-collaborative optimization of Shared energy storage

(c) Collaborative optimization of factory station storage

(d) Collaborative optimization of Shared energy storage

Figure 5: Double side storage and discharge in collaborative and non-coordinated mode

III. C. 2) Comparison of simulation results of scheduling strategies

By analyzing the energy scheduling and demand response scheduling, it can be concluded that the algorithm proposed in this paper can effectively reduce the operating cost, customer dissatisfaction and CO2 emission simultaneously. The scheduling strategies of other algorithms cannot simultaneously optimize the multi-objective values under their respective objective weights.

A comparison of the simulation results of the scheduling strategies is presented in Table 2, which shows the compromise scheduling results for the three objectives on one day. The total operating cost, total dissatisfaction and emissions obtained by the proposed algorithm are lower than the other methods. For example, the running cost, customer dissatisfaction, and pollution emission in summer are 3052.518, 4.416, and 4064.249, respectively.

Although the total customer dissatisfaction of the MOPSO algorithm is 6.585% lower than that of the SSA algorithm in winter, the running cost and CO2 emission of the SSA algorithm are 5.193% and 31.98% lower than that of the MOPSO algorithm.

Table 2: Comparison of scheduling strategy simulation results

Season	Target function	SSA	MOPSO	GBDT	Deep residual network
Summer	Operating cost	3052.5184	4290.5485	5984.4865	3884.8345
	User dissatisfaction	4.4159	5.2896	7.9485	7.0468
	Pollution discharge	4064.2485	7163.4852	10786.8698	5516.9485
Winter	Operating cost	4896.6648	5164.8695	6548.5368	5453.6345
	User dissatisfaction	4.8596	4.5396	8.4569	8.8495
	Pollution discharge	7923.5948	11648.9544	12953.4648	9315.7645

IV. Conclusion

Microgrid load forecasting shows good performance in practical applications. By comparing with GBDT and deep residual networks, the deep learning model proposed in this paper achieves the minimum prediction error in several microgrids. In particular, the model achieves the lowest MAPE value of 2.0485% in Microgrid 1, and the performance is more stable in Microgrid 2 with a MAPE value of 5.5169%. Although the prediction error is larger in Microgrid 3 and Microgrid 4, the model is still able to provide more accurate load forecasts overall. In future applications, the model can provide effective support for real-time scheduling and optimization decision-making in microgrids, especially when dealing with sudden load fluctuations and uncertainties, which can achieve more accurate load allocation. In addition, the introduction of the model in this paper not only improves the accuracy of load forecasting, but also enhances the intelligence level of the microgrid management system by synthesizing different deep learning techniques, which provides technical support for the efficient use of renewable energy.

References

[1] Saleh, M., Esa, Y., & Mohamed, A. (2018). Applications of complex network analysis in electric power systems. Energies, 11(6), 1381.

[2] Singh, S., Gautam, A. R., & Fulwani, D. (2017). Constant power loads and their effects in DC distributed power systems: A review. Renewable and Sustainable Energy Reviews, 72, 407-421.

[3] Afolabi, O. A., Ali, W. H., Cofie, P., Fuller, J., Obiomon, P., & Kolawole, E. S. (2015). Analysis of the load flow problem in power system planning studies. Energy and Power Engineering, 7(10), 509-523.

[4] Adefarati, T., Bansal, R. C., & Justo, J. J. (2017). Reliability and economic evaluation of a microgrid power system. Energy Procedia, 142, 43-48.

[5] AL-Nussairi, M. K., Bayindir, R., Padmanaban, S., Mihet-Popa, L., & Siano, P. (2017). Constant power loads (cpl) with microgrids: Problem definition, stability analysis and compensation techniques. Energies, 10(10), 1656.

[6] Sahebkar Farkhani, J., Zareein, M., Najafi, A., Melicio, R., & Rodrigues, E. M. (2020). The power system and microgrid protection—A review. Applied Sciences, 10(22), 8271.

[7] Tah, A., & Das, D. (2016). Operation of small hybrid autonomous power generation system in isolated, interconnected and grid connected modes. Sustainable Energy Technologies and Assessments, 17, 11-25.

[8] Muhtadi, A., Pandit, D., Nguyen, N., & Mitra, J. (2021). Distributed energy resources based microgrid: Review of architecture, control, and reliability. IEEE Transactions on Industry Applications, 57(3), 2223-2235.

[9] Wang, T., O'Neill, D., & Kamath, H. (2015). Dynamic control and optimization of distributed energy resources in a microgrid. IEEE transactions on smart grid, 6(6), 2884-2894.

[10] Nti, I. K., Teimeh, M., Nyarko-Boateng, O., & Adekoya, A. F. (2020). Electricity load forecasting: a systematic review. Journal of Electrical Systems and Information Technology, 7, 1-19.

[11] Xu, L., Li, C., Xie, X., & Zhang, G. (2018). Long-short-term memory network based hybrid model for short-term electrical load forecasting. Information, 9(7), 165.

[12] Imani, M. (2021). Electrical load-temperature CNN for residential load forecasting. Energy, 227, 120480.

[13] Jalali, S. M. J., Ahmadian, S., Khosravi, A., Shafie-khah, M., Nahavandi, S., & Catalão, J. P. (2021). A novel evolutionary-based deep convolutional neural network model for intelligent load forecasting. IEEE Transactions on Industrial Informatics, 17(12), 8243-8253.

[14] Kohn, W., Zabinsky, Z. B., & Nerode, A. (2015). A micro-grid distributed intelligent control and management system. IEEE Transactions on Smart Grid, 6(6), 2964-2974.

[15] Dhifli, M., Lashab, A., Guerrero, J. M., Abusorrah, A., Al-Turki, Y. A., & Cherif, A. (2020). Enhanced Intelligent Energy Management System for a Renewable Energy-Based AC Microgrid. Energies, 13(12), 3268.

[16] Wen, L., Zhou, K., Yang, S., & Lu, X. (2019). Optimal load dispatch of community microgrid with deep learning based solar power and load forecasting. Energy, 171, 1053-1065.

[17] Zhao Zilong,Tang Jinrui,Liu Jianchao,Ge Ganheng,Xiong Binyu & Li Yang. (2022). Short-term microgrid load probability density forecasting method based on k-means-deep learning quantile regression. Energy Reports,8(S5),1386-1397.

[18] Yin Yufeng,Wang Wenbo & Yu Min. (2024). Short-Term Load Forecasting of Microgrid Based on TVFEMD-LSTM-ARMAX Model. Transactions on Electrical and Electronic Materials,25(3),265-279.

[19] Muhammad Sajid Iqbal,Muhammad Adnan,Salah Eldeen Gasim Mohamed & Muhammad Tariq. (2024). A hybrid deep learning framework for short-term load forecasting with improved data cleansing and preprocessing techniques. Results in Engineering,24,103560-103560.