Long- and Short-Term Power Load Forecasting Model Based on the Fusion of Multiple Deep Learning Algorithms

Shuoqin Lin, Zhengsheng Zhan, Shukai Cheng, Canjian Huang, Dangyue Lai

School of Automation Science and Engineering, South China University of Technology, Guangzhou, China

Keywords: Electric Load; ARIMA-NAR; LSTM; PSO-XGBoost; Double Carbon

Abstract: Against the background of global energy transition and "double carbon" target, power system load forecasting is crucial in ensuring energy security, optimizing resource allocation and promoting sustainable economic and social development. In order to cope with the complex changes of power system load, improve the prediction accuracy, and provide accurate support for grid operation and scheduling, the establishment of scientific and effective load forecasting models has become a key research topic. In this paper, based on the NAR model and LSTM model improved by ARIMA and CEEMD algorithms, the short-term and long-term power load forecasting methods are proposed respectively, and the PSO-XGBoost algorithm is used to construct the electric load fitting model to study the abnormal sudden changes in the power load data. Using the constructed model, the maximum and minimum values of daily loads of each industry in the region for the next three months are predicted and the prediction accuracy is analyzed.

1. Introduction

In the context of global energy transition and the pursuit of carbon neutrality, accurate load forecasting in power systems is essential to ensure energy security, optimize resource allocation, and support sustainable economic and social development. To provide accurate support for grid operation and dispatch decisions, it is necessary to cope with the complex dynamic changes of power system loads and improve the forecast accuracy. Therefore, the development of robust and effective load forecasting models has become a top research priority.

2. Literature References

2.1 Model Assumptions

Stable operation of the regional power system: it is assumed that the regional power system will be able to operate stably without major faults or outage events during the forecast period. This will ensure the reliability of the input data for the forecast model.

Load data is cyclical and trending: Observation of the data reveals that the daily load data shows obvious cyclical changes, such as higher loads in the morning and evening peak hours and lower loads at night. At the same time, there may be a certain trend in the load data over time, for example, with economic development, the load data may show an upward trend.

Economic Activity Remains Stable: It is assumed that the economic activity in the region remains stable during the forecast period and there will be no major economic fluctuations. This will help the forecasting model to more accurately capture the impact of economic activity on electricity demand

2.2 ARIMA modeling

The prediction accuracy of the model built with the preprocessed data will be significantly improved compared to the model built with the original data[1]. In this data cleaning, we remove vacant samples and duplicates, and we do not remove outliers because we want to analysis the mutation of noise. Then start building the ARIMA model.

The ARIMA model is a statistical model for time series analysis and forecasting. It combines autoregressive (AR), difference (I) and moving average (MA)[2] components to capture

  DOI: 10.25236/iwmecs.2024.012

autocorrelation and trend characteristics in time series.

Among them, the current value of AR is determined by the linear combination of the values of the previous periods; I is used to smooth the non-stationary time series; and the current value of MA is determined by the linear combination of the prediction errors of the previous periods, and the related formula is as follows:

$$y_t = c + \sum_{i=1}^{p} \Phi_i y_{t-i} + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + \epsilon_t \tag{1}$$

In the formula, c is the constant term, Φ_i is the autoregressive coefficient, θ_j is the moving average coefficient, and ϵ_t is the error term.

Data smoothness is the key prerequisite when using ARIMA model for time series modeling. As can be seen from Figure 2, the initial data are not smooth. At this point, the trend component in the data is successfully eliminated by performing a differencing operation (i.e., calculating the difference between neighboring data points) on the time series, which makes the mean and variance of the series stable, thus satisfying the smoothness requirement of the ARIMA model. The range of data fluctuations in the differenced data (e.g., Fig. 3) becomes more concentrated, which not only reduces the long-term trend and seasonal fluctuations in the time series, but also maintains the autocorrelation characteristics of the original data and ensures that the model accurately predicts the future moments.

Figure 1 shows the 15-minute load data for a region, and each point represents the total active power of the charge load in the region during this 15-minute period. The analysis shows that the total active power in the region fluctuates up and down around a certain value over time with a certain periodicity. The fluctuation of total active power over time is large, so it can be determined that the sequence is a non-stationary sequence, which needs to be differentiated for subsequent analysis. In order to minimize the data loss caused by differencing and to guarantee the accuracy of the subsequent modeling, first-order differencing is performed on the charge load data in the region. It was therefore determined that the parameter d in ARIMA(p,d,q) was initially determined to be 1.

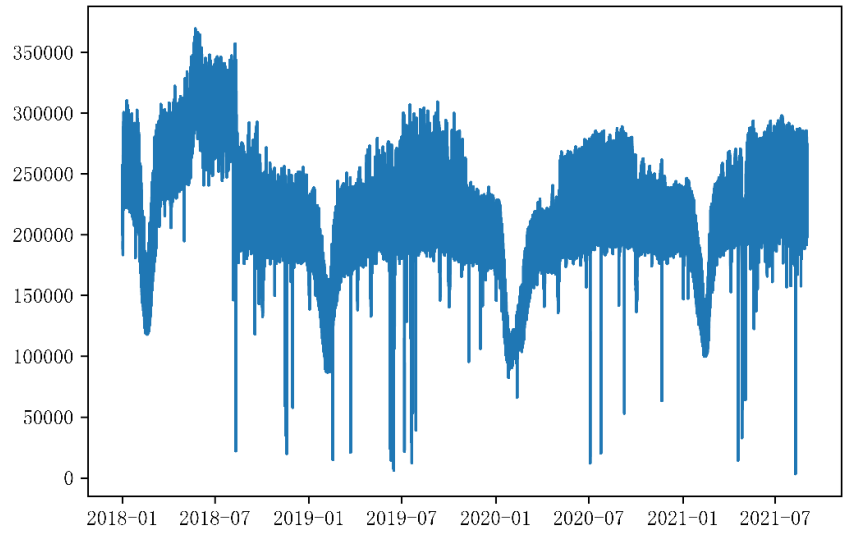

Fig. 1 Raw timing diagram

The ARIMA model requires the series to satisfy smoothness, and differencing is mainly used to eliminate data fluctuations by subtracting the previous value from the value of the data, making the data smooth. Assuming that the total active power data after differencing the charge load data in the region is an unsteady time series, the ADF test of the differenced data using SPASS yields a p-value of 0.01 (<0.05) for the Q-statistic after differencing, according to which the original hypothesis can be rejected, and the series is considered as a smooth time series.

In addition, this bit also visualizes the results of the 15-minute load data differencing of the region's electric charge, and the following figure shows a plot of the results after first-order differencing of the raw data, with the vertical coordinate being the power differencing value and the horizontal

coordinate being the temporal parameter, which corresponds to the temporal order of the data

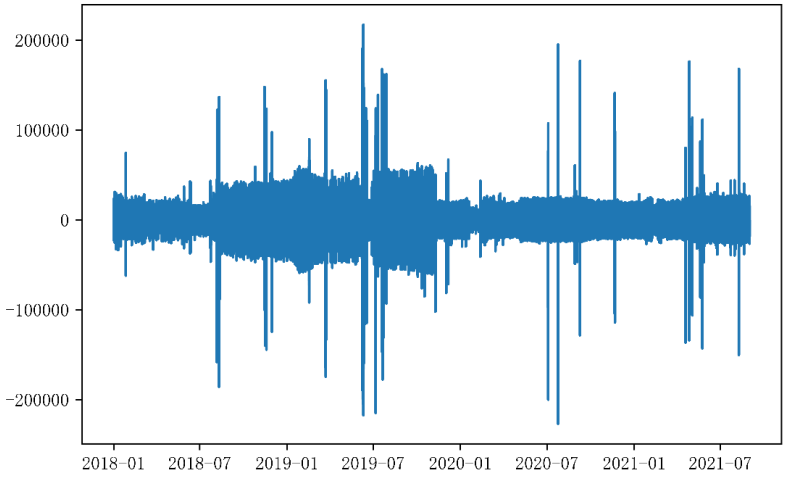

Fig. 2 Timing diagram after differencing

Analyzing the time series plot after differencing, it can also be seen that the new to data has less fluctuation over time, tends to be stable, and is a relatively stable time series. In order to find the most appropriate ARIMA model parameters, the following will combine the autocorrelation plot of the differenced series, the partial autocorrelation plot and the AIC criterion to determine the parameters p and q in the ARIMA (p,d,q) used. the following figure shows the autocorrelation and partial autocorrelation plots of the differenced series:

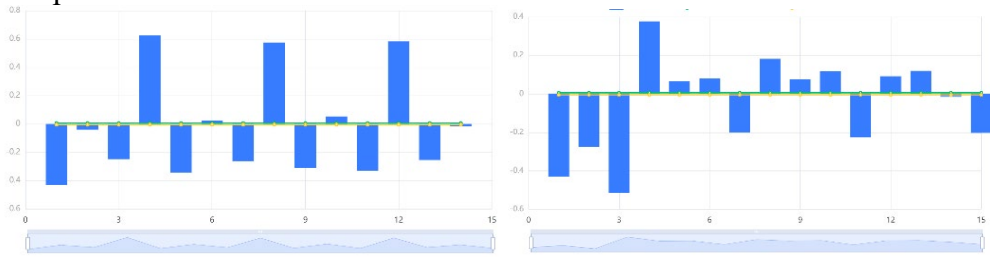

Fig. 3 ACF (left) and PACF (right) of differential sequences

The autocorrelation plot of the difference series shows that there is a significant autocorrelation at the first order lag, but the autocorrelation decays rapidly as the lag order increases, and then the autocorrelation coefficients in most of the lags approach zero, which implies that the smoothness of the series has been significantly improved.

In the partial autocorrelation plot of the difference series, there is significant partial autocorrelation only in the first few lags (especially the first lag), while the partial autocorrelation coefficients of the rest of the lags are mostly close to zero. The partial autocorrelation plots reflect the fact that the difference treatment effectively removes the long-term dependence from the series, making the series more suitable for ARIMA fitting and forecasting.

After determining the smoothness of the data, this paper tries several ARIMA models and selects the best model through the AIC criterion, and the results are shown in Table 1.

Table 1. Residual analysis of several optimal models with different ARIMA's AIC values

Model structure	AIC	Ljung-Box	Jarque-Bera	Heteroskedasticit
ARIMA(2,1,2)	inf	-	-	-
ARIMA(0,1,0)	2789090	-	-	-
ARIMA(1,1,0)	2763033	-	-	-
ARIMA(0,1,1)	2746882	-	-	-
ARIMA(1,1,1)	**2745730**	**9.52**	**5341743**	**1.25**
ARIMA(2,1,1)	2746335	-	-	-
ARIMA(1,1,2)	Inf	-	-	-
ARIMA(0,1,2)	2747657	-	-	-

From Table 1, it can be seen that the ARIMA(1,1,1) model performs best among several candidate models with the lowest AIC value of 2745730, therefore, ARIMA(1,1,1) is selected as the final model in this paper, and the related fitting results are given in Figure 6. Meanwhile, the fitting effect of ARIMA(1,1,1) model Ljung-Box test shows that the residuals of the model are still autocorrelated, while the Jarque-Bera test shows that the distribution of the residuals deviates from the normal distribution, which may affect the accuracy of the prediction. In addition, the heteroskedasticity of the residuals may also adversely affect the robustness of the model. Therefore, although the ARIMA(1,1,1) model performs better under the AIC criterion, the NAR model will be introduced to further optimize the model to improve the accuracy and stability of the predictions.

MAE represents the mean absolute error between the predicted value and the actual value, and the value is 7793.22 kw. Compared with the magnitude of the original data, the MAE is small, which indicates that the fitting accuracy of the ARIMA model is better and within the acceptable range, and the residual part will be processed next to make the model more refined, and the actual fitting results are shown in Fig. 4.

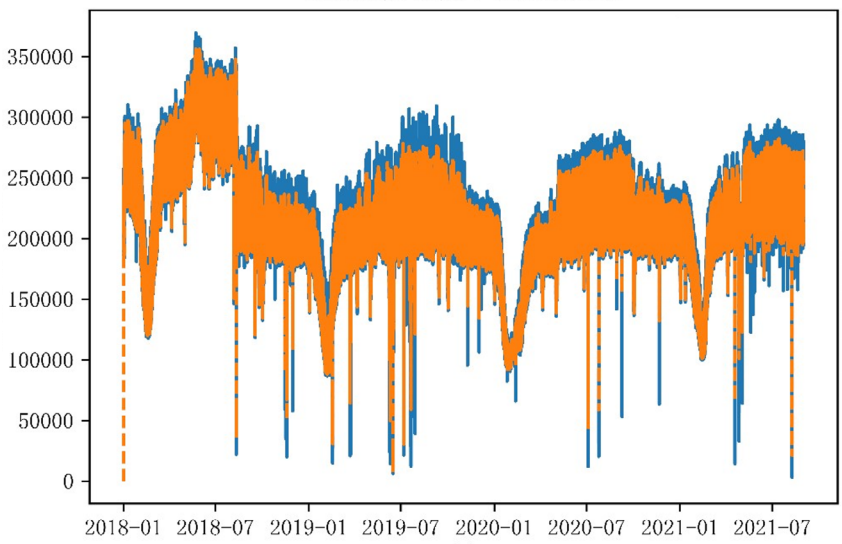

Fig. 4 ARIMA model fitting results

2.3 Residual decomposition based on CEEMD modeling

The original residual sequence usually contains multiple frequency components - a high-frequency component, a mid-frequency component, and a low-frequency component. Using the NAR model directly to process the entire residual sequence, the model may have difficulty in capturing the complex dynamic behavior in the different frequency components at the same time.The CEEMD decomposition separates the complex residual signal into different intrinsic modal functions (IMFs), with each IMF denoting a relatively simple frequency component, and the mathematical formula can be written as follows:

$$r(t) = \sum_{i=1}^{n} IMF_i(t) + R(t) \tag{2}$$

where $r(t)$ represents the residual time series, $IMF_i(t)$ is the ith eigenmode function, and $R(t)$ is the residual.

In order to improve the modeling and prediction of the NAR model, CEEMD (Complete Ensemble Empirical Modal Decomposition)[3] is performed prior to the use of the NAR (Nonlinear Autoregressive) model to decompose the complex, nonlinear, and nonsmooth time series signal into multiple relatively simple, and distinguishable signal components. By separating the signal components at different frequencies, the NAR model can be modeled separately for each IMF, which makes it easier to capture the nonlinear patterns in each frequency component, thus improving the overall prediction accuracy.

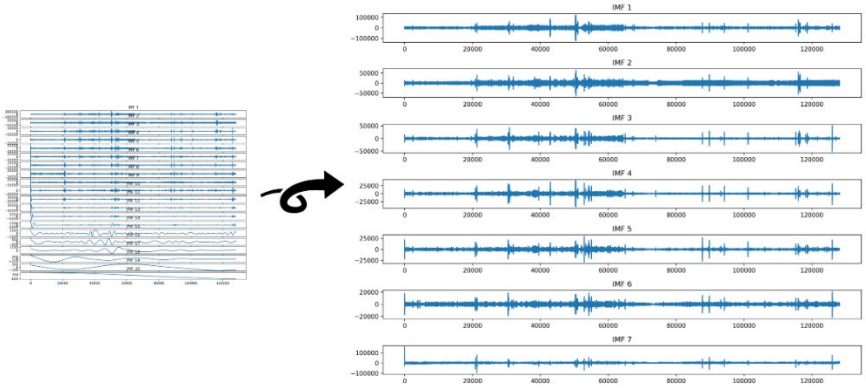

Fig. 5 CEEMD residual decomposition plot

The original residual dataset is decomposed by adding white noise, and the results of each decomposition are visualized. As can be seen from Figure 5, when the number of decompositions reaches 20 times, the decomposed data shows a set monotonicity, i.e., the nonlinear components in the residuals are all eliminated.

2.4 Residual prediction based on NAR model

Nonlinear Autoregressive (NAR) model is a neural network model suitable for time series forecasting. The nonlinear mapping of the neural network can be realized by multiple levels of activation functions (e.g., ReLU, Sigmoid, etc.). For a given time series, the NAR model recursively predicts future values based on previous predictions by iterating. It predicts future values by constructing a nonlinear functional relationship between past observations of the time series.

By CEEMD decomposition, the original residual sequence $r(t)$ is decomposed into several intrinsic modal functions $IMF_i(t)$ and residual terms $R(t)$. Each IMF represents a different frequency component that can be viewed as a smoother signal. Next, we construct a NAR model for each IMF separately to perform nonlinear prediction.

$$IMF_i(t+1) = f_i\big(IMF_i(t), IMF_i(t-1), \ldots, IMF_i(t-p)\big) + \epsilon_i \tag{3}$$

where $f(\cdot)$ is the NAR model corresponding to the ith IMF and ϵ_i is the noise term.

The training process of the NAR model requires determining several key parameters and optimizing the performance of the model through a reasonable training method. The ultimate goal of model training is to improve the prediction accuracy for each IMF signal by minimizing the prediction error.

The autoregressive order p is how many past historical point-in-time values are used as inputs by the NAR model when making future point-in-time predictions. In order to balance the use of more historical information as well as introduce the risk of overfitting, p is set to 10 in this case.

The number of neurons in the hidden layer determines the capacity of the neural network, i.e., the level of nonlinear complexity that the model can fit. More neurons means that the network has a higher capacity and can learn more complex nonlinear mappings, but it also increases the training difficulty and the risk of overfitting, and the number of neurons in the hidden layer is set to 10 for overall consideration.

The choice of the activation function of the NAR network will greatly affect the error of the current layer. Because the data set is relatively large, if the sigmoid function or tanh is used as the activation function here, it may lead to the current layer output derivative values are not greater than 1, making the gradient disappear in the iterative process, leading to overfitting conditions in the final training. Therefore, using the ReLU function as the activation function and choosing the neuron whose derivative value is 1 for learning can effectively avoid the gradient disappearance, and the model training is completed successfully.

The nonlinear residuals extracted by CEEMD are imported into the established NAR model, and the data are randomly disrupted and divided into the training set and validation set in the ratio of 7:3 to start training. Here, the maximum number of training rounds is set to 50, and the model training

error converges at the 43rd training, at which time the error loss on the training set is 1.4%; the error var-loss on the test set is 0.89%, indicating that the model training results can be well generalized to the validation set.

The predicted residual results of NAR are summed with the ARIMA model results to obtain the final prediction results, as shown in Fig. 6:

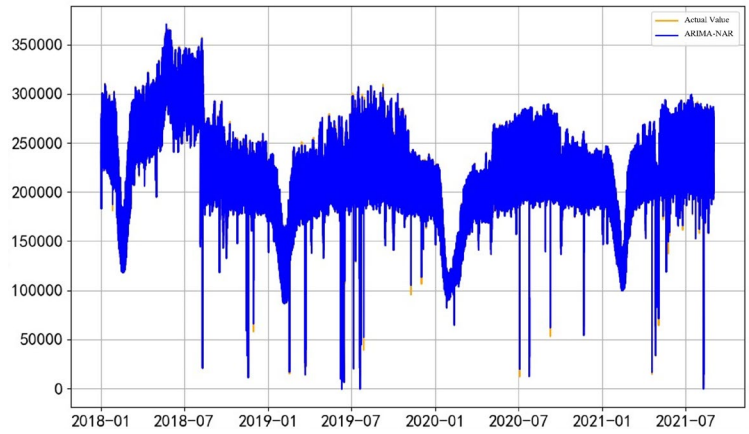

Fig. 6 ARIMA-NAR model fitting results

Combined with the fitting result plots, it can be found that the ARIMA-NAR model has a better fit to the actual values compared to the ARIMA model, and its prediction will be more accurate, Figure 7 shows the ARIMA-NAR prediction result plots for the 15-minute charge load of the region for the next 10 days:

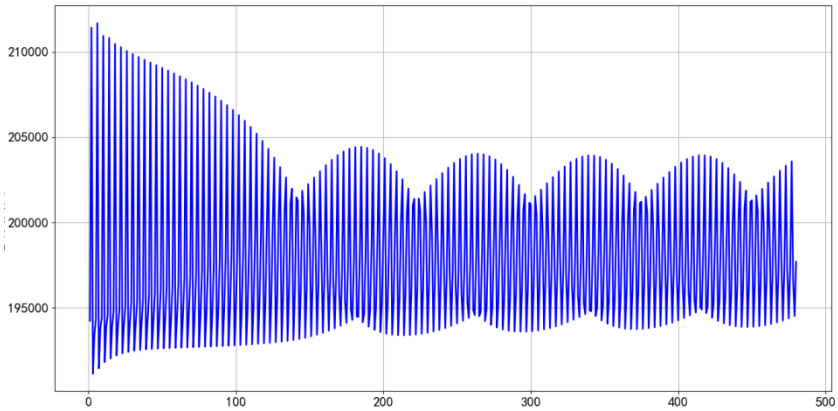

Fig. 7 Final projected results

Analyze the results of the next 10-day interval 15-minute charge load data prediction, you can see that the power system load will fluctuate up and down around 200,000kw or so, and the overall load has a tendency to decline first and then remain stable. In addition can also first power system in a short period of time power load consumption of total active power oscillates violently, with the time change fluctuation is obvious, but for a long time there is a certain periodicity, indicating that the future period of time the power system as a whole power to maintain a relatively stable state.

2.5 Long-term electricity load forecasting based on ARIMA-CEEMD-LSTM model

XGBoost[4] is based on the existing gradient boosting tree such as GBDT, lambdaMART and other boosting tree algorithms algorithms improved at the system and algorithmic level, through the PSO and other algorithms for parameter optimisation[5], the core of the optimisation of the objective function value, compared to the LSTM is more in line with the problem studied in this paper.

Long Short-Term Memory (LSTM) networks are a special type of Recurrent Neural Network (RNN) that excel at learning and remembering long sequences.LSTMs are designed to exploit their unique gating properties to overcome the limitations of traditional RNNs, especially the gradient vanishing and explosion problems, which impede the learning of long term dependencies in sequence

data. The main constituent structures of an LSTM cell are the forgetting gate f_t, the input gate i_t, and the output gate o_t. These gates control the flow of information inside and outside each LSTM cell. In addition, each LSTM cell maintains a cell state C_t that serves as a memory for relevant information across time steps. Oblivion gates determine which information from the previous cell state C_{t-1} should be discarded. The output value of the oblivion gate is between 0 and 1, where 1 indicates complete retention of information and 0 indicates complete oblivion.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{4}$$

where f_t is the activation vector of the forgetting gate; σ is the sigmoid function; W_f is the weight matrix of the forgetting gate; h_{t-1} is the previous hidden state; and b_f is the bias term of the forgetting gate.

The input gate determines what new information will be stored in the unit state. It consists of two parts: the input gate layer and the tanh layer, which is responsible for creating new candidate values and adding them to the cell state.

$$i_t = \sigma(W_i \cdot [h_{t-1}] + b_i) \tag{5}$$

$$\tilde{C}_t = \tan\mathrm{h}(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{6}$$

Where W_i and W_C are the weight matrices of the input gates and candidate cell states, respectively; b_i and b_C are the bias terms; and $\tilde{C}_t$ is the candidate cell state created by the tanh layer. And the cell state C_t is a combination of the old cell state and the new candidate cell state. The forgetting gate controls the retention of the old state and the input gate controls the integration of the new information.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{7}$$

where C_{t-1} is the old cell state; and the operator denotes the multiplication of the corresponding elements of the matrix.

The output gate determines which information about the current cell state will be passed on to the next hidden state h_t, which in turn is used as input for the prediction and the next time step.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{8}$$

$$h_t = o_t \cdot \tan\mathrm{h}(C_t) \tag{9}$$

where W_o is the weight matrix of the output gate; b_o is the bias term of the output gate.

Typically, an LSTM model consists of multiple LSTM layers followed by one or more fully connected layers (Dense Layer) to generate the final prediction. The input to the model is the serialized data and the output is the corresponding predicted values. The input data is first processed by defining the original sequence $F_O = \{f_1, f_2, \cdots, f_n\}$. After that, this sequence is utilized to divide the training set $F_{tr} = \{f_1, f_2, \cdots, f_m\}$ and test set $F_{te} = \{f_{m+1}, f_{m+2}, \cdots, f_n\}$. At the same time, the constraints $m < n$ and $m, n \in N$ should be satisfied.When it involves the difference in magnitude and order of magnitude between different features, the data in the dataset are normalized to obtain the normalized training set $F'_{tr} = \{f'_1, f'_2, \cdots, f'_m\}$.

In order to adapt to the characteristics of the input of the hidden layer, the data segmentation method is applied to F'_{tr}, and the length of the segmentation window is set to L. The input of the model after segmentation is $X = \{X_1, X_2, \cdots, X_L\}$, and the corresponding theoretical output is $Y = \{Y_1, Y_2, \cdots, Y_L\}$. Input X into the hidden layer, which contains L isomorphic LSTM cells connected by before and after moments. Set the cell state vector size to S_{state}.

The trained LSTM network (denoted as $LSTM^*_{net}$) is applied for prediction. The prediction process uses an iterative approach. Initially, the last line of the theoretical output is Y set to $F_f = \{f'_{m-L+1}, f'_{m-L+2}, \cdots, f'_m\}$. Input F_f into $LSTM^*_{net}$ to output the result, then merge the last L-1 data points of Y_f and P_{m+1} into a new row of data.

$Y_{f+1} = \{f'_{m-L+2}, f'_{m-L+3}, \cdots, f'_{m+1}\}$. After that, Y_{f+1} is input into $LSTM^*_{net}$ to get the prediction result P_{m+2} at time m+2 yes. Following this iterative approach, the prediction sequence $P_O =$

$\{P_{m+1}, P_{m+2}, \cdots P_n\}$.

If the original data is normalized, in order to convert the prediction results back to the original magnitude and scale, the z-score score is used to denormalize, denoted as:

$$P_{te} = de_{zscore(P_o)} = \{P^*_{m+1}, P^*_{m+2}, \cdots, P^*_n\} \tag{10}$$

where P^*_k is calculated as follows:

$$P^*_k = P_k \times \left\{ \sqrt{\frac{\sum_{t=1}^{n}\left(f_t - \frac{\sum_{t=1}^{n} f_t}{n}\right)^2}{n}} + \frac{\sum_{t=1}^{n} f_t}{n} \right\} \tag{11}$$

The formula calculates the final prediction taking into account the standard deviation and mean of the original series.

The exact accuracy of the model can be determined by comparing the fitted series F_{te} with the predicted series P_{te}. Mean absolute error, root mean square error, or other relevant metrics can be used to quantify the accuracy of the model.

For this question, the topic requires forecasting the maximum and minimum values of daily loads for the next three months, which is a typical time series data with time dependence and periodicity.LSTM is excellent in capturing nonlinear relationships and long- and short-term dependencies in time series, while ARIMA excels in dealing with linear trends and periodic components. Meanwhile, grid load data may be affected by a variety of factors, such as weather and season. This complexity and multiplicity makes it difficult for a single model to fully capture the patterns in the data.LSTM is able to handle high-dimensional input data and nonlinear relationships, while ARIMA can effectively handle linear components. Therefore, the combination of these two models can better capture the global characteristics of the data.

The residuals after CEEMD decomposition from 5.3.2 are used as the original dataset for LSTM. Since the dimension of the given data is only 1, there is no need to normalize and denormalize the data. The training and test sets were set to be 70% and 30%, respectively, and a total of 500 iterations were performed.The initial learning rate of the LSTM was set to be 0.1 and reduced by multiplying the learning rate by a factor of 0.2 after 250 iterations to prevent overfitting of the model. The segmentation window length was taken as 3 and the number of hidden units was set to 100. The data was imported into MATLAB and then computed.

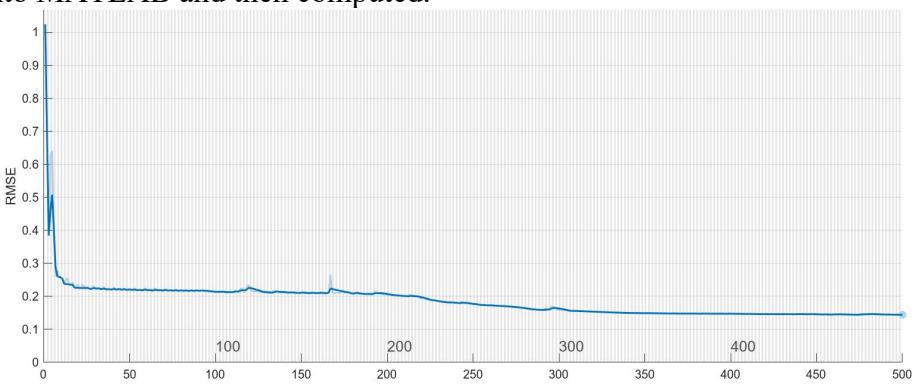

Fig. 8 Plot of RMSE with number of iterations

As can be seen in Figure 8, the RMSE of the LSTM model converges at 500 iterations. After the maximum number of iterations is reached, the model is trained. The final RMSE value is 4920.In order to exclude the influence of the original data magnitude on the RMSE magnitude, we normalize this RMSE by the maximum and minimum values. The normalization formula is:

$$RMSE^* = \frac{RMSE - MIN(Y)}{MAX(Y) - MIN(Y)} \tag{12}$$

Where $MAX(Y)$ and $MIN(Y)$ refer to the maximum and minimum values of the original training data, respectively. The normalized $RMSE^*$ is low from 11.65% to 7.63%, which indicates that the model improved by ARIMA-CEEMD model is better trained.

The residuals of the loads at 15-minute intervals for the next three months were predicted using the trained model, and the residuals were added with the ARIMA prediction to obtain the final prediction results:

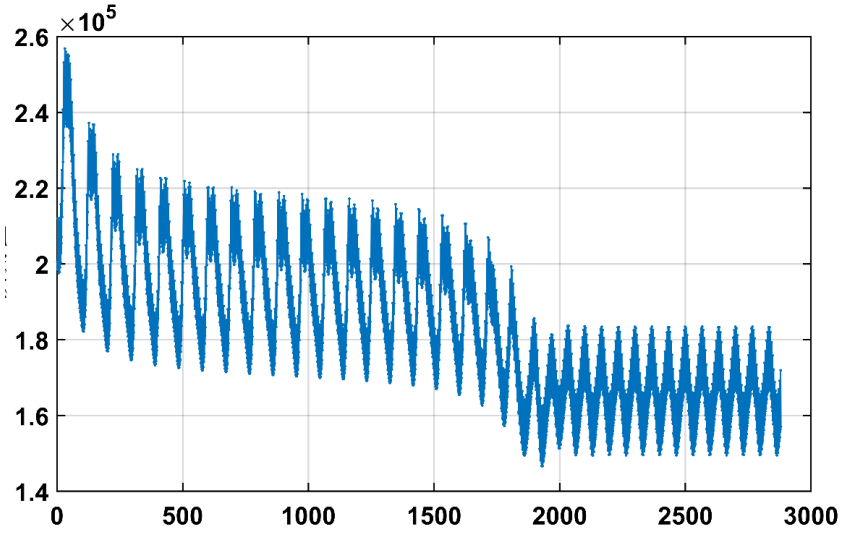

Fig. 9 ARIMA-LSTM final prediction result plot

From the analysis of Fig. 9, it can be seen that the regional charge load in this region for the next three months shows a strong cyclicity, with an overall fluctuation around 200,000kw up and down, and a decreasing trend in the local daily charge load. The integration of the data yields the total value of daily charge load in the region for the next three months, which is presented in Table 2:

Table 2 Regional daily electricity and load for the next three months

Date	AIC
1	20940066.19
2	19642111.25
3	19119044.93
…	…
28	15492933.32
29	15784551.51
30	15435657.74

3. Summary

The regional power prediction model constructed in this paper excavates the characteristics of the power load in time distribution through a preliminary analysis of the electricity load of the regional power grid. On the basis of the traditional time series prediction ARIMA model, the concept of non-linearization of residuals is introduced, and the data mining of short-term and long-term power loads is carried out by NAR and LSTM respectively, and the final prediction results are obtained by combining them with the linear part. In addition, based on the PSO-XGBoost model, a regression model between the electricity load of various industries and the main meteorological data was constructed. The model is adaptable and flexible. The model can make relatively accurate predictions of electricity consumption in the future based on the past electricity load data of the region, and combined with local weather conditions, it can help various industries formulate better power supply strategies and help achieve the "dual carbon" goal

References

[1] Almuhaideb, S., Menai, M.E.B. Impact of preprocessing on medical data categorization. Front. Comput. Sci. 10, 1082-1102 (2016).

[2] HU Jianbo,LU Zhipeng,LI Feng. Forecasting China's carbon emission intensity under the goal of “peak carbon” - an analysis based on LSTM and ARIMA-BP models[J]. Financial Science, 2022(02):89-101.

[3] Hu G, Wang M. Survey on Biomedical Signal Processing[J]. Journal of Data Acquisition & Processing, 2015, 30(5): 915-932

[4] Zhang P, Jia Y, Shang Y. Research and application of XGBoost in imbalanced data. international Journal of Distributed Sensor Networks. 2022;18(6).

[5] H. Jiang, Z. He, G. Ye and H. Zhang, “Network Intrusion Detection Based on PSO-Xgboost Model,” in IEEE Access, vol. 8, pp. 58392-58401, 2020