Internet of Things User Behavior Analysis Model Based on Improved RNN

Keling Bi*
School of Information Engineering, Liaodong University, Dandong 118003, China

***Abstract*—Currently, there are issues with low efficiency and outdated Internet of Things resource allocation. To study real Internet of Things user behavior data, a Bayesian optimization algorithm is used to automatically select hyperparameter combinations and construct an Internet of Things user behavior analysis model based on long short-term memory. The results showed that the prediction accuracy of the model reached 96.8% and 97.53% on the training and validation sets, while in the set 50 maximum iterations, the model achieved 80.78% on the test set. In comparing the performance between the research model and the traditional recurrent network model, it was found that the optimal prediction accuracy of the research model was 80.78%, which was better than the comparison model. The application results of the research model in short-term power load forecasting also indicated that the prediction accuracy of the Internet of Things user behavior analysis model based on the improved recurrent network has reached a good level, far superior to the comparative model. The results have important application value for allocating energy and resources in Internet of Things systems.**

***Keywords*—Internet of Things; user behaviors; recurrent neural network; Bayesian optimization; long short-term memory; hyperparameter**

I. Introduction

The continuous progress of science and technology has led to an exponential growth trend in the amount of information, and the Internet of Things (IoT) technology has also been further developed. In recent years, the further popularization of 5G technology, the further reduction of hardware costs, and the development of big data technology have led to a significant advancement in the IoT. At this stage, the IoT is no longer merely a network of objects and systems; it has evolved into a comprehensive system that connects all the people, processes, and resources involved, forming a seamless and automated interaction process [1-2]. At the current stage of development, the IoT integrates the interactive factors related to people. The data types and quantities involved in the whole IoT system have greatly increased, and the traditional processing system technology has been unable to meet the actual needs of [3-4]. At this time, artificial intelligence (AI) technologies have emerged as the optimal avenue for industrial advancement within the IoT domain. Furthermore, the integration of AI has become pervasive across various production and operational processes. However, the practical application of AI technology in the IoT often has the problem of poor timing [5-6]. Given this, many scholars have researched the analysis of IoT user behavior, and methods such as mathematics, data mining, and machine learning (ML) have emerged [7-8]. Mathematical methods are characterized by high complexity, limited interpretability, and greater difficulty in achieving results. In contrast, data mining and simple calculation technology are relatively straightforward, yet they often prove ineffective in predicting user behavior. Based on this, the long and short-term memory (LSTM) recurrent neural network (RNN) method in the ML method is used to construct the prediction model. At the same time, to solve the problems of high complexity and optimization difficulty of ML methods, the Bayesian optimization algorithm (BOA) is studied based on the prediction model of LSTM-RNN, which can realize the automatic optimization of the prediction model. This study first hopes to find out the correlation and the implied behavior mode of the user behavior data of the IoT, aiming to provide more temporal data for more accurate, fast, and convenient IoT services. Secondly, the study is expected to construct a user behavior prediction model for the IoT that can adapt to different environmental characteristics. This will provide a basis for user behavior information that can be used to improve the quality of system services in the IoT and further improve the efficiency of resource allocation in the system. Finally, the study validates the model through training in terms of losses, effects, and load forecasting. Through these verification methods, the performance and practical application of the proposed method can be effectively analyzed.

II. Literature Review

In the context of IoT research pertaining to user behavior analysis (UBA), to promote the application of big data in various fields, Hou et al. proposed an IoT unstructured big data analysis online client algorithm built on ML algorithm. The algorithm was used in other big data analysis scenarios and verified to be more efficient than other comparative algorithms [9]. Abdelmoumin et al. conducted research on the performance of IoT-based anomaly-based intelligent intrusion detection system (IDS) ML model. Compared with deep learning-based IDS, anomaly-based ML-based IDS exhibited lower performance and prediction accuracy (PA) in detecting intrusions in IoT [10]. Xu et al. proposed a data-driven intrusion and anomaly detection method using IoT automatic ML to address the current issues of IoT network attacks and intrusions. This algorithm technology not only saved the computational cost of runtime test data, but also solved a multi-class classification problem with an accuracy of 99.7%, with significant advantages over existing algorithms [11]. To solve the optimal pricing and bundling problem of ML-based IoT services, Alsheikh et al. defined data value and service quality from the perspective of ML and proposed an IoT market model. Compared to independent sales, bundling IoT services could maximize the profits of service providers [12]. Woźniak et al.

analyzed the network traffic of various IoT solutions based on deep learning models to address network security issues in network physical systems. The results confirmed that even when the number of evaluated network features decreased, the model was very effective in identifying potential threats, with an accuracy rate of over 99% [13]. Zhao et al. proposed an IoT intrusion detection method based on lightweight deep neural networks. On two real NID datasets, this method had good classification performance, lower model complexity, and smaller model size, making it suitable for IoT traffic classification in both normal and attack scenarios [14].

In studies employing LSTM networks for behavioral prediction, the LSTM network will also be utilized to conduct such studies based on the most abundant user purchase and social behavior data available on the Internet. For example, Sakar et al. used a multi-layer perceptron for feature classification and trained a LSTM-RNN to predict the probability of the user leaving the current site. The purpose was to take corresponding measures to improve the purchase conversion rate of the website [15]. LSTM networks are also used in some prediction studies of action and behavioral trajectories. To estimate the movement intention of people in intelligent manufacturing service of human-robot cooperation, Liu et al. used LSTM network to extract the time pattern of human movement and automatically output the prediction results before the movement. This approach was taken to ensure the efficiency and safety of the system [16]. Huang et al. proposed three properties of asymmetric driving behavior and constructed a vehicle following model based on a LSTM-RNN [17]. Yimin et al proposed a human-centered trajectory tracking control strategy, using a LSTM network to design a model predictive control method considering insertion vehicle driver behavior to track the reference trajectory [18].

In conclusion, many scholars have achieved notable advancements in the enhancement of IoT-UBA performance, economic value, and network security. However, existing IoT-UBA research focuses on predicting whether a certain behavioral action will occur, without considering the usage characteristics of items in IoT and the relationships between users and items. In addition, existing research only considers the sequence relationship between user behavior occurrence, that is, using pre-order behavior to analyze the possible behaviors that may occur in the post-order. This merely indicates the potential for future behavior without specifying the timing of subsequent actions. Consequently, it is unable to enhance the precision, speed, and accessibility of IoT services or provide more detailed temporal data. Based on this, this study innovatively proposes the use of BOA for model hyperparameter selection, improves RNN, and constructs an IoT-UBA model that can adapt to different environmental characteristics. The model can provide more time-series data for more accurate, fast, and convenient IoT services. Furthermore, it can enhance the efficiency of resource allocation within the system.

III. Methods and Materials

This study first conducts relevant data mining and preprocessing based on real IoT user behavior data. Subsequently, the user behavior dataset is employed to train an IoT-UBA model founded upon LSTM-RNN, and BOA is utilized to automatically select model hyper-parameter combinations.

A. Research Data Mining and Preprocessing

The dataset used contains sensor activation data from 00:00:00 to 23:59:59 every second within 30 days in an IoT environment for TWO family members. 86,400s of data are generated every day, with a total of 2,592,000 pieces of data included in the 30 day period. The activation of each sensor represents the use of a certain item, so the activation data of sensors can to some extent represent the behavior of residents in the IoT environment. Table I shows the location, type, and ID of the sensors.

In Table I, the dataset folder "ARAS" contains two folders: "HouseA" and "HouseB", representing two different households. Each household folder contains 30 text files in the format "DAY_x" and one "Readme" text file to explain the basic information of data, sensor information, and activity instructions. Before mining and further processing data, the first is to compress and merge the data. Considering that the IoT-UBA model constructed using RNN is used, this study uses a total of 30 days of monthly data for merging and compression. After conducting a series of pre-experiments, the dataset compressed in units of 20s performs the best in three candidate scenarios: 10s, 20s, and 30s. Therefore, this study will compress the data in units of 20 seconds, meaning that every 20 rows of data will be merged into one row. The merged data in this row will take the maximum activation data of all sensors within 20 seconds, that is, all activated sensors within 20 seconds will be marked as "1". Subsequently, this study formats the data based on the time step determined during the data mining auto-correlation analysis process. The time window processed sequence dataset is shown in Fig. 1.

Dataset $\{x_1, x_2, ..., x_n\}$ is processed into the format shown in Fig. 1 using a time window of length t. The n sequence data are processed into n-t+1 sequence data with a time step of 1. Among them, in the data used, size is to be determined as the hyper-parameter of the model, and the input data dimension is 20 dimensions. The time step is determined by auto-correlation analysis to be 60, which means that the behavior of IoT users in the first 60 time units will have an impact on the current IoT user behavior. The original data consist of 129,600 20 dimensional sensor activation data, with a data format of (129,600*20). After processing through a time window of 60 in length, it has been transformed into a 3D dataset in the shape of (129,541*60*20). Among them, 129,541 is calculated by (129,600-60+1).

TABLE I. ID, TYPE, AND LOCATION OF SENSORS

Sensor ID	Sensor Type	Place
Ph1	Photocell	Wardrobe
Ph2	Photocell	Convertible Couch (Used as bed for Resident 2)
lr1	IR	TV receive
Fol	Force Sensor	Couch
Fo2	Force Sensor	Couch
Di3	Distance	Chan
Di4	Distance	Chair
Ph3	Photocell	Fridge
Ph4	Photocell	Kitchen Drawer
Ph5	Photocell	Wardrobe
Ph6	Photocell	Bathroom Cabinet
Co1	Contact Sensor	House Door
Co2	Contact Sensor	Bathroom Door
Co3	Contact Sensor	Shower Cabinet Door
So1	Sonar Distance	Hall
So2	Sonar Distance	Kitchen
Di1	Distance	Tap
Di2	Distance	Water Closet
Te1	Temperature	Kitchen
Fa3	Force Sensor	Bed

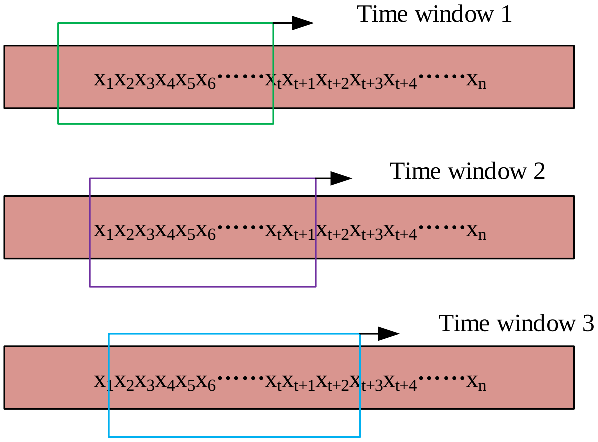

Time window 1
$X_1X_2X_3X_4X_5X_6\cdots\cdots X_tX_{t+1}X_{t+2}X_{t+3}X_{t+4}\cdots\cdots X_n$
Time window 2
$X_1X_2X_3X_4X_5X_6\cdots\cdots X_tX_{t+1}X_{t+2}X_{t+3}X_{t+4}\cdots\cdots X_n$
Time window 3
$X_1X_2X_3X_4X_5X_6\cdots\cdots X_tX_{t+1}X_{t+2}X_{t+3}X_{t+4}\cdots\cdots X_n$

Fig. 1. Time window processed sequential dataset.

The data used in this study is relatively large (greater than 10,000 but less than 100,000), so the data are segmented into a 60% training set, a 20% validation set, and a 20% testing set. After format transformation, 129,541 pieces of data will be divided into the first 77,727, middle 25,907, and last 25,907 pieces according to the time series for model construction. Among them, 77,727 pieces of data are used for the preliminary construction and optimization of the model, and 25,907 pieces of data are used to verify the model's generalization ability during the optimization process. After the model construction is completed, the generalization ability of the final model is tested using the last 25907 pieces of data.

B. Building an IoT-UBA Model Based on Improved RNN

In solving problems related to time series data, complete and continuous strings, images, etc., there is a certain degree of correlation between the front and back data of the dataset involved. That is to say, in model training, the data cannot only enter the neural network unilaterally and independently for parameter training . To build a better IoT-UBA model, it is necessary to fully consider the information carried by the pre-order data in the process of drawing conclusions. Among them, RNN is used to solve sequence related problems. In RNN, the parameter W used for information transmission in hidden states is the same. In the backpropagation gradient descent method used to solve parameters, the chain rule of differentiation will result in the solved gradient containing the multiplication of weights. When the weight value is greater than or less than 1, multiplication will cause the gradient to expand infinitely or approach zero infinitely. The former is called gradient explosion, while the latter is called gradient disappearance. The gradient explosion problem can be solved using gradient truncation, which limits the maximum value of gradients [19-21]. Similarly, if the gradient vanishing problem is to be solved by restricting the minimum value of the gradient, it will result in the obtained weights not reflecting the actual impact of the node well. This study proposes to use LSTM network to solve this problem, and its network structure is shown in Fig. 2.

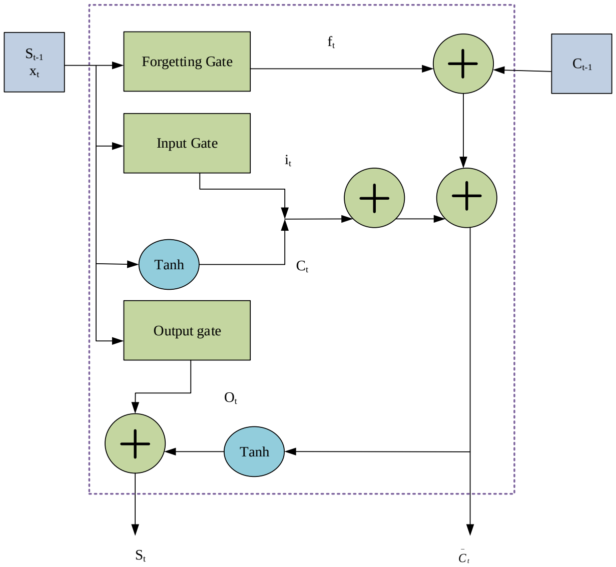


Fig. 2. Internal structure of LSTM unit.

The LSTM unit has three parts: input gate, output gate, and forget gate. The activation function corresponding to the three control gates is the sigmoid function. The output of sigmoid is between 0 and 1. The activation function can be understood as a control valve in the layman's sense, where "0" represents the control valve being closed and the information being completely filtered. "1" represents the control valve closing, and the information is completely retained. By controlling the opening and closing of the valve, important information can be filtered. If the x_t, the previous unit state S_{t-1}, and the long-term state (LTS) C_{t-1} that was memorized in the previous time are input, the unit is entered first and the forget gate is entered next. At this point, a vector f_t composed of numbers is obtained, as shown in Eq. (1).

$$f_t = \sigma\left(W_f \times S_{t-1} + U_f + x_t + b_f\right) \tag{1}$$

In Eq. (1), $f_t \in [0,1]$. W_f, U_f, and b_f represent the weights of the forget gate. The multiplication of f_t and the LTS C_{t-1} from the previous moment determines how much information in C_{t-1} enters C_t, as shown in Eq. (2).

$$K_t = f_t \otimes C_{t-1} \tag{2}$$

In Eq. (2), K_t represents the need to retain the information of the LTS C_t at this moment, and this result will be used as one of the inputs to the input gate. In the input gate, x_t and S_{t-1} are processed by a tanh function to obtain the information $\bar{C}_t$ of the LTS to be added, as shown in Eq. (3).

$$\bar{C}_t = \tanh\left(W_C \times S_t - 1 + U_C + x_t + b_C\right) \tag{3}$$

In Eq. (3), W_C, U_C, and b_C represent the weights of long-term unit states, similar to forgetting gates. x_t and S_{t-1} will also pass through a signoid function to obtain a vector i_t composed of numbers. This vector is multiplied by $\bar{C}_t$ to determine how much information needs to be added to the LTS C_t at this moment, as shown in Eq. (4).

$$i_t = \sigma\left(W_i \times S_{t-1} + U_i + x_t + b_i\right) \tag{4}$$

In Eq. (1), $i_t \in [0,1]$. W_i, U_i, and b_i are the weights of the input gate. By combining the calculation results of the forget and input gates, the LTS C_t is obtained, as shown in Eq. (5).

$$C_t = K_t + i_t \otimes \bar{C}_t \tag{5}$$

In Eq. (5), C_t is only transmitted within the network and will be passed to the next unit as one of the inputs to the next unit. Similarly, passing through the signoid function will generate a numerical vector o_t, as shown in Eq. (6).

$$o_t = \sigma\left(W_o \times S_{t-1} + U_o + x_t + b_o\right) \tag{6}$$

The LTS C_t at this moment is processed by the tanh function and multiplied by o_t to determine how much information in the LTS C_t is output as the unit state S_t at that moment, as shown in Eq. (7).

$$S_t = o_t \otimes \tanh\left(C_t\right) \tag{7}$$

Among them, the unit state S_t on the last time step is the final output of the model. After each batch of data is transmitted into the network, parameters are updated through gradient descent to reduce the loss between actual and predicted values. This study uses binary cross entropy as the loss function M, as shown in Eq. (8).

$$M = -\frac{1}{20}\sum_{I=1}^{20}\left[y_i \log\left(\bar{y}_i\right) + \left(1 - y_i\right)\log\left(1 - \bar{y}_i\right)\right] \tag{8}$$

In Eq. (8), y represents the numerical vector of the true value. $\bar{y}$ represents the numerical vector of the predicted value. In relatively fixed and bounded usage scenarios like IoT, user behavior interacts with relatively fixed objects within a certain range, with limited influencing factors and following certain patterns. To this end, an IoT-UBA model is constructed based on LSTM, as shown in Fig. 3.

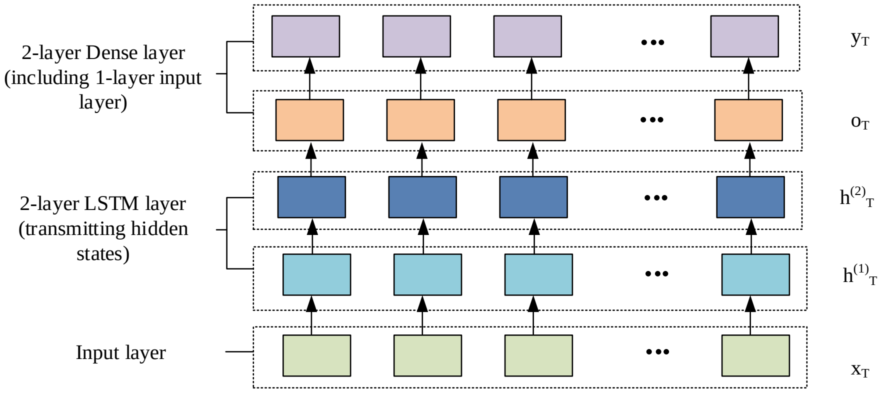

2-layer Dense layer (including 1-layer input layer)
2-layer LSTM layer (transmitting hidden states)
Input layer
•••
yT
oT
h(2)T
h(1)T
xT

Fig. 3. Schematic diagram of the research model.

This model is a neural network model consisting of two LSTM layers, one Dense layer, and one output layer. On the data of each time step t (t∈[1,T])) with a total of T time steps, the hidden state $h_t^{(1)}$ (t∈[1, T]) of the first layer LSTM layer and the hidden state $h_t^{(2)}$ (t∈[1,T])) of the second layer LSTM layer will be passed to the next LSTM layer at the next time step. Moreover, the hidden state includes LTS C_t^i (i∈[1,2],t ∈[1,T]) and short-term state S_t^i (i∈[1,2],t∈[1,T]), thereby achieving the effect of memorizing the hidden information in the preceding data. In addition, this study relies on experience to select hyperparameters that may have a better analytical effect on the model (Table II).

TABLE II. SELECTION OF MODEL HYPERPARAMETERS

Hyper-parameter	Value	Type
Lstm_layer	2	int
Activation_lstm	Tanh	string
Lstm_output_dim	110	int
Dense_layer	2	int
Activation_dense	Softsign	string
Activation last	Sigmoid	string
Drop_out	0.2	float
Batch_size	64	int
Nb_epoch	4	int
Optimizer	Rmsprop	string
Loss	Binary cross entropy	string

C. Optimization of Hyperparameters Based on BOA

After constructing an IoT-UBA model based on LSTM, it is found that having too many hyperparameters can lead to an increase in model complexity and easily lead to overfitting. To solve overfitting problems and improve model generalization ability, ML engineers generally adjust hyper-parameter combinations based on experience to achieve a better level of generalization [22]. However, manual parameter tuning is difficult to fully consider all relevant influencing factors and historical performance. Even though time and manpower are spent manually adjusting parameters and achieved good results, the constructed prediction model only performs well in the IoT-UBA corresponding to specific datasets. The predictive ability of this hyperparameter combination among more IoT users is still unknown [23]. This study proposes using BOA to construct an adaptive hyperparameter selection algorithm, selecting personalized hyperparameter combinations that perform best for different users. The purpose of hyperparameter optimization (HPO) in ML is to find the hyper-parameters of a given ML, which returns the best performance measured on the validation set. Unlike model parameters, hyperparameters need to be set before training. The HPO equation is shown in Eq. (9).

$$\lambda = \arg\min_{\lambda \subseteq x} f(\lambda) \tag{9}$$

In Eq. (9), λ represents the hyperparameter and $f(\lambda)$ represents the minimum objective score evaluated on the validation set, which is also the loss that needs to be optimized for the model. BOA combines the advantages of manual parameter tuning with manual experience and grid search, as well as automatic selection through random search, to track past evaluation results. It forms a probability model by using these results, seeking a combination of hyperparameters in the probability model that can minimize losses. Before using automatic optimization algorithms to search for the optimal combination of hyperparameters, it is necessary to define the domain space for hyperparameter search. This study constructs a large-scale hyperparameter domain space based on the research model. HPO requires defining the objective of BOA optimization, which is the loss of the objective function. The ultimate goal of selecting hyperparameter combinations in this study is to improve the model's generalization ability. The greater the generalization ability of the model, the better, but the loss of the BOA function can only be the minimum value, so the loss value Q of BIA is shown in Eq. (9).

$$Q = 1 - Pa \quad (9)$$

In Eq. (9), Pa represents the PA of the test set. To learn the "black box" between hyperparameter combinations and optimization objectives, this study uses tree-structured Parzen estimator (TPE) to construct optimization algorithms. The TPE algorithm can construct a probability model for optimization objectives by combining existing hyperparameters and their corresponding loss values. Through this probability model, it is possible to find the hyperparameter combination that minimizes the optimization objective and maximizes the probability. Then, based on the selected hyperparameter combination, the trained model updates the input-output pairs and continues to construct a new probability model. Cycling the above process until the loss value reaches its lowest point, and thus completing the construction of the IoT-UBA model based on improved RNN. The construction process of the model is shown in Fig. 4.

The research model construction in Fig. 4 first involves data mining and preprocessing based on real IoT user behavior data. Then, a prediction model needs to be constructed based on LSTM network, and hyperparameters need to be selected based on experience. Finally, BOA is used to automatically select model hyperparameter combinations.

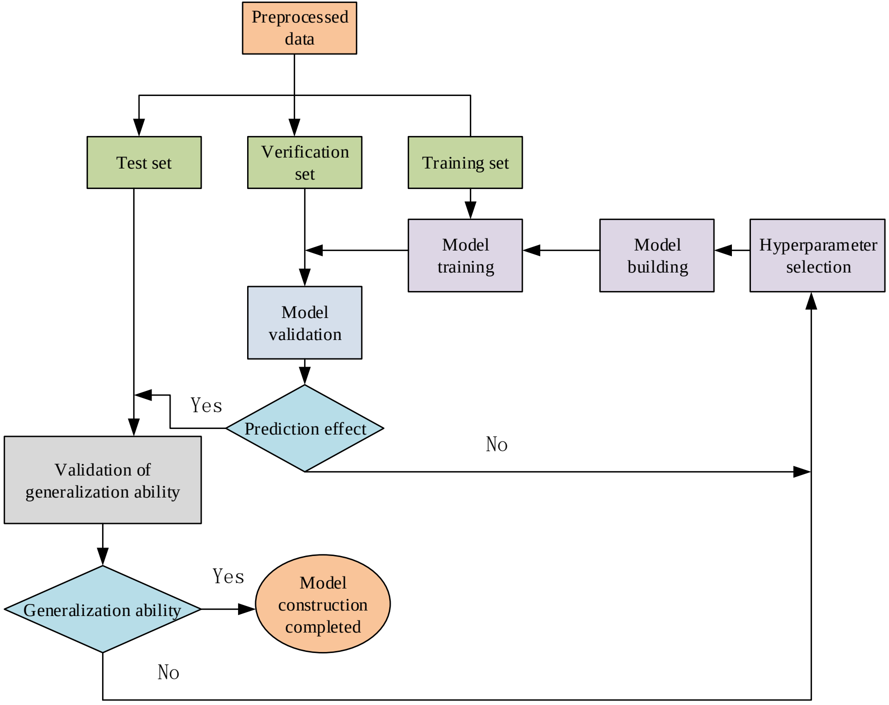


Fig. 4. Model construction process.

IV. Results

A. Training and Validation of Research Models

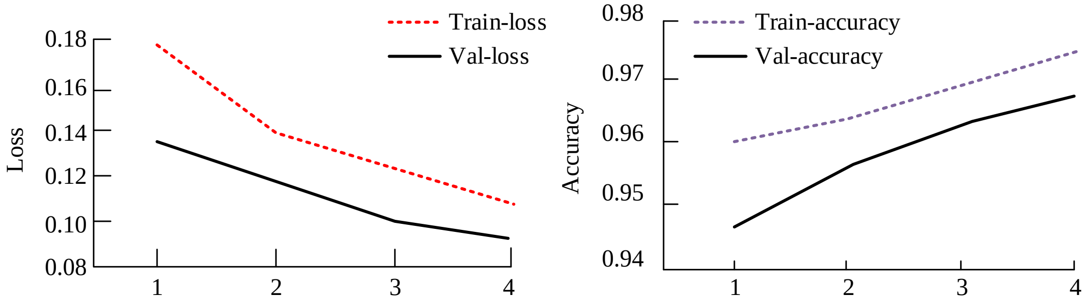
To test the model effectiveness, this experiment uses the processed training and validation sets for model training and validation and uses the test set to test the model's generalization ability. Table III shows the relevant experimental environments.

This experiment is conducted using XiaoXinPro 14ITL 2021, with a processor environment of 11th GenIntel(R)Core (TM)i5-1135G7@2.40GHz-2.42GHz, a memory capacity of 16.0GB, and the operating system of Windows10. The software used in data mining is SPSS Modeler18.0. The programming environment involved is Python 3.8.3 and the programming IDE used is Anaconda3. Python is used for data preprocessing, model construction, and optimization algorithms. The divided training set and validation set are put into the constructed model. Based on research data and model characteristics, binary classification accuracy is used as a measure of model PA and binary classification cross entropy is adopted as a loss function. The training results of the research model are displayed in Fig. 5.

In Fig. 5, as the training iteration progresses, the loss of the research model on the training and validation sets decreases continuously. At the completion of the last epoch of training, the loss on the training set decreases from the initial 0.1800 to 0.1084, and on the validation set from the initial 0.1362 to 0.0915. The learning ability of the model continues to improve. The PA of the model on both the training and validation sets surpasses 90%, and as the iteration process continues to rise, the final accuracy reaches 96.8% and 97.53%. The dataset, hyperparameter domain space, and objective function are input into BOA, and the hyperparameter combinations selected for each iteration of BOA, their corresponding training time, and loss values are recorded for tracking optimization history. The optimal combination of hyperparameters for localization prediction is shown in Fig. 6.

TABLE III. EXPERIMENTAL OPERATING ENVIRONMENT

Parameter	Experimental Environment
Tool	XiaoXinPro 14ITL 2021
Processor	11th GenIntel(R)Core (TM)i5-1135G7@2.40GHz-2.42GHz
Memory capacity	16.0GB
Operating system	Windows10
Data mining software	SPSS Modeler18.0
Programming environment	Python3.8.3
Programming IDE	Anaconda3
model building	Python3.8.3



(a) Model loss (b) Model accuracy

Fig. 5. Model training results.

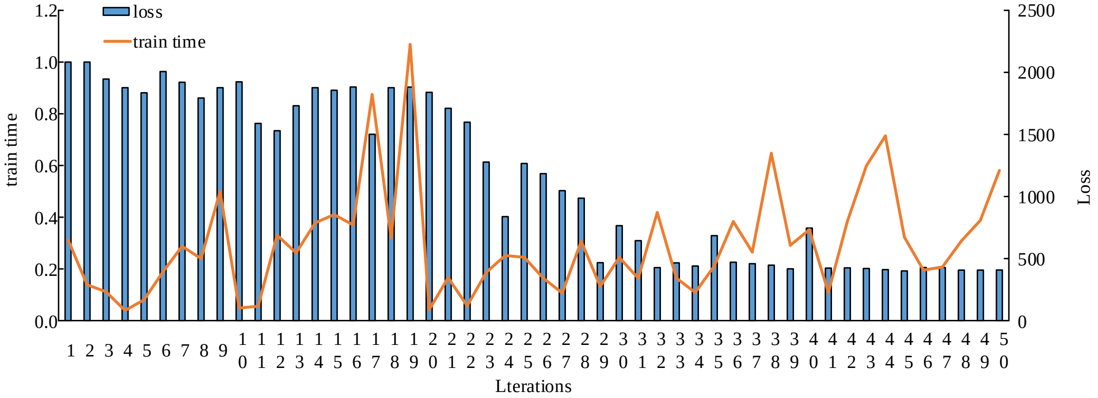
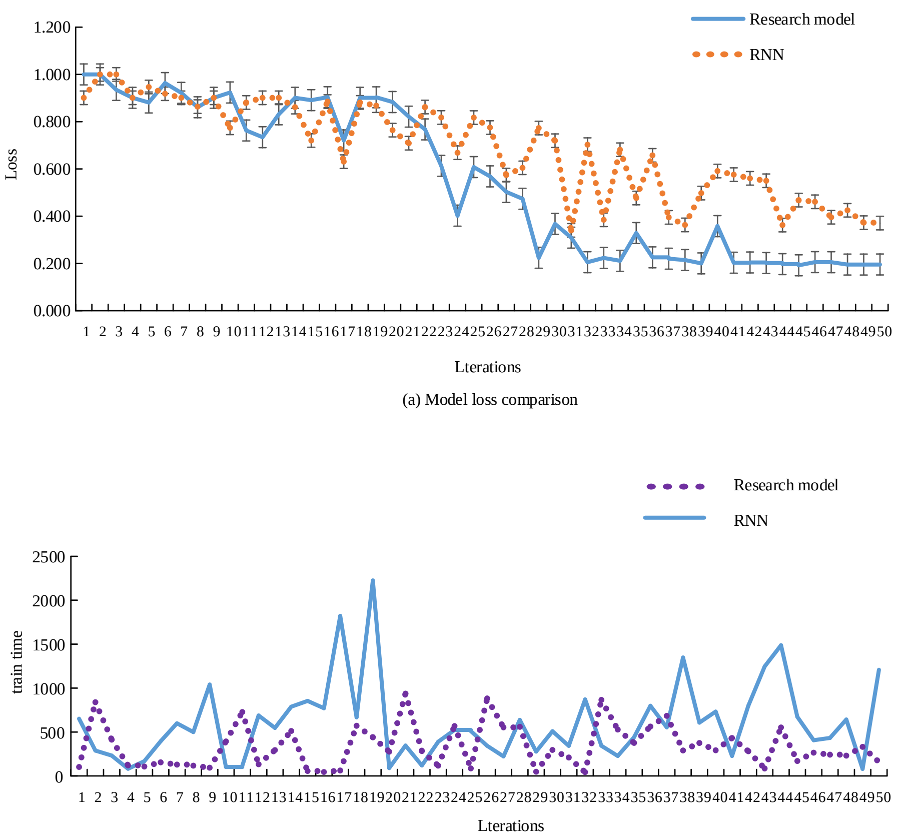


Fig. 6. Iteration loss and training time.

In Fig. 6, the loss shows a significant downward trend with the number of iterations, indicating that when selecting hyperparameter combinations, BOA will choose hyperparameters that are more likely to perform better based on the probability model of the loss with respect to hyperparameters. In the set maximum of 50 iterations, the loss of the objective function reaches its minimum value in the 45th iteration, and the optimal model loss trains under the corresponding hyperparameter combination conditions in this iteration is 0.19. The model's PA on the test set reaches 80.78%, showing that BOA enhances its generalization ability, and searches for a model hyperparameter combination with better prediction ability based on the user behavior characteristics in the IoT environment. To further validate the improved RNN in IoT-UBA, a similar neural network model is constructed using traditional RNN, and BOA is also used to select the optimal hyper-parameter combination. The experimental results are shown in Fig. 7.



(a) Model loss comparison

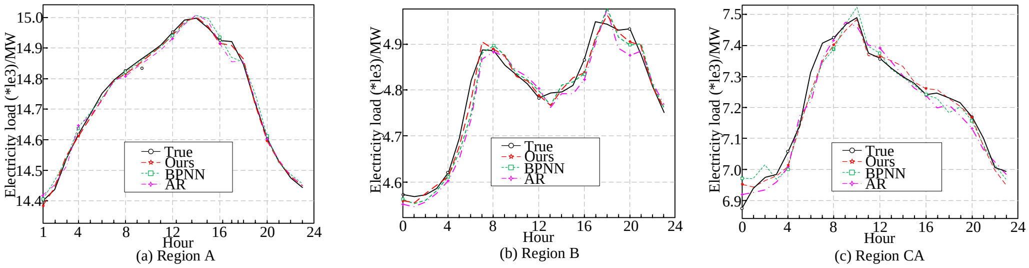
(b) Model train time comparison

Fig. 7. Comparison of research model and RNN model effects.

Fig. 7 (a) shows the comparison of the loss value changes between the traditional RNN model iteration process and the research model. Both models show a spiral downward trend as the iterations increase. Among them, the minimum loss value of the traditional RNN is 0.34, which means its best PA is only 66.01%, lower than the 80.78% of the research model. Fig. 7 (b) shows the comparison of training time between LSTM and RNN models. The research model reaches the minimum loss value in the 31st iteration, while RNN only selects the optimal hyper-parameter combination in the 45th iteration. The average training time for the research model is 338 seconds, and this value for the RNN model reaches 614 seconds. In terms of iteration times and training time, the research model is significantly better than the RNN model.

B. *Application of Research Models in Short-term Power Load Forecasting*

To test the practicality of the IoT-UBA built on improved RNN, this study selects two benchmark models, auto-regression model (AR) and back propagation neural network (BPNN), to validate the proposed improved RNN model. This study selects three different datasets, namely three load datasets provided by official websites in three foreign regions. The first one is the electricity load data of region A from February 1, 2023 to August 31, 2023, with a sampling rate of 1 hour and a sample size of 5112. The second one is the electricity load data of region B for the whole year of 2023, with a sampling rate of 1 hour and a sample size of 8760. The third one is the electricity load data of region C for the whole year of 2023, with a sampling rate of 1 hour and a sample size of 8760. The power load data of these three regions are processed through data preprocessing and the processed data are divided into datasets. The daily load forecasting results of each model on three different datasets are displayed in Fig. 8.



(a) Region A

(b) Region B

(c) Region CA

Fig. 8. Daily load forecasting results of each model on three different datasets.

From Fig. 8 (a), in region A, the prediction effect of each model is overall and the load prediction curve in the figure almost coincides with the actual situation. However, Fig. 8 (a) shows that the prediction effect of the research model is better than that of other comparison models. In particular, in the two periods of 13:00 and 17:00, the error rate of the research model is only 0.13% and 0.09%, which is small and has more accurate PA than the comparison model. In the data set of Fig. 8 (b) in region B, in general, at the very beginning, each model has a good prediction effect. However, with the increase of mean absolute percentage error (MAPE), the deviation between the prediction of various models and the reality are increasing, which leads to the decrease of the accuracy of the prediction. Especially in the period of 20:00, each model prediction deviation reaches the largest. The AR model, BPNN model, and acting model of daily load prediction error rates reach 12.13%, 5.72%, and 1.39%, respectively. Among them, the research model error rate is lower than other contrast model. The prediction results and the actual load change trend are the most consistent and the performance effect is better. In region C of Fig. 8 (c), in general, the prediction effect of each model is not good. With the increase of MAPE, the deviation between the prediction of various models and the reality are increasing, leading to a decrease in the accuracy of the prediction. Among them, the period of 2:00 appears the largest prediction error. In the 17:00 period, the AR model reaches the largest error in the prediction. In both 2:00 and 17:00, the prediction effect of the research model is relatively excellent. This indicates that although traditional power system modeling methods can describe the overall trend of the power system, as the complexity of the system increases, the operational efficiency of the power system also decreases, resulting in huge economic losses to the power system. The root mean square error (RMSE) and MAPE of load forecasting for each model within seven days are exhibited in Fig. 9.

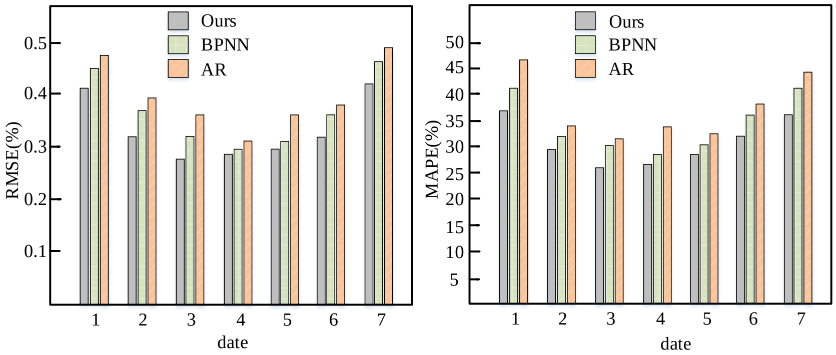


Fig. 9. Comparison of weekly load prediction errors among different models.

In Fig. 9, the weekly load PA of the IoT-UBA is much better than that of BPNN and AR, and BPNN is higher than AR. The comparison between its predicted load value and the actual load value shows that through more accurate prediction of user behavior, the prediction of load value has completed a good level.

V. Discussion

As the IoT technology becomes increasingly pervasive in all aspects of people's daily lives, the collection of user data within IoT systems will become more comprehensive. The data accurately reflect the behavioral patterns and living habits of users. Providing better services and timely responses to users, effectively utilizing this information to predict user behavior and needs, and systematically improving management and service quality are key to enhancing the competitiveness and service level of various industries in the future. The proposed method sets a maximum of 50 iterations. At 45 iterations, the objective function has the smallest loss, and the optimal model has a loss of only 0.19 (test set), with an accuracy of 80.78%. The results of this study are consistent with the predictions of the lightweight dense random neural network proposed by Latif et al. [24] for IoT intrusion detection, and the model performs well. This is because when selecting appropriate inputs for the research model, BOA selects a more promising hyperparameter based on loss estimation. This can search for super parameter combinations with higher predictive performance based on user behavior characteristics in IoT scenarios, thereby enhancing the performance of the model. In this study, the model lost the smallest loss in 31 cycles, the best PA was 80.78%, and the learning time was 338s, which showed better performance. However, the accuracy of the industrial IoT detection method based on graph neural network proposed by Wu et al. [25] can only reached 79.71%. This is because the hyperparameters have been optimized, shortening the iteration and learning time of the model, thereby improving its generalization ability. In summary, the analysis shows that using BOA to select model hyperparameter combinations further improves the generalization ability of the prediction model, enabling the model to construct adaptive prediction models based on the characteristics of different users in different IoT environments.

VI. Conclusion

In response to the current problems in the application of IoT technology, this study utilized BOA to construct an IoT-UBA model based on an improved RNN that can adapt to different environmental characteristics. The results demonstrated that the PA of the research model on both the training and validation sets was greater than 90%, and as the iteration process continued to increase, the final accuracy reached 96.8% and 97.53%. In 50 maximum iterations, the model achieved a PA of 80.78% on the test set. In the comparison of the performance between the research model and RNN, the minimum loss value of traditional RNN was 0.34, which meat its best PA was only 66.01%, lower than the 80.78% of the research model. The research model reached the minimum loss value in the 31st iteration, while RNN only selected the optimal hyper-parameter combination in the 45th iteration. The average training time for the research model was 338 seconds, and this value for the RNN model reached 614 seconds. In terms of iteration times and training time, the research model was significantly better than the RNN model. The application results of the research model in short-term power load forecasting also indicated that the PA of the IoT-UBA model based on improved RNN has reached a good level, far superior to other comparative models. However, this study is based on a sufficient amount of historical user behavior data. In practical use, the IoT-UBA model will inevitably face the situation of insufficient historical user behavior data at the initial startup of the system. In the future, in-depth research will be conducted in this area.

References

[1] Sarker I H, Khan A I, Abushark Y Bl. Internet of things (iot) security intelligence: a comprehensive overview, machine learning solutions and research directions. Mobile Networks and Applications, 2023, 28(1): 296-312.

[2] Liu Y, Wang J, Li J. Machine learning for the detection and identification of Internet of Things devices: A survey. IEEE Internet of Things Journal, 2021, 9(1): 298-320.

[3] Saheed Y K, Abiodun A I, Misra S. A machine learning-based intrusion detection for detecting internet of things network attacks. Alexandria Engineering Journal, 2022, 61(12): 9395-9409.

[4] Ullah A, Anwar S M, Li J. Smart cities: The role of Internet of Things and machine learning in realizing a data-centric smart environment. Complex & Intelligent Systems, 2024, 10(1): 1607-1637.

[5] Zhou H, She C, Deng Y. Machine learning for massive industrial internet of things. IEEE Wireless Communications, 2021, 28(4): 81-87.

[6] Saharkhizan M, Azmoodeh A, Dehghantanha A. An ensemble of deep recurrent neural networks for detecting IoT cyber attacks using network traffic. IEEE Internet of Things Journal, 2020, 7(9): 8852-8859.

[7] Khalil R A, Saeed N, Masood M. Deep learning in the industrial internet of things: Potentials, challenges, and emerging applications. IEEE Internet of Things Journal, 2021, 8(14): 11016-11040.

[8] K. Bhosle and V. Musande.Evaluation of Deep Learning CNN Model for Recognition of Devanagari Digit. Artif. Intell. Appl.2023,1(2):114-11.

[9] Hou R, Kong Y Q, Cai B. Unstructured big data analysis algorithm and simulation of Internet of Things based on machine learning. Neural Computing and Applications, 2020, 32(10): 5399-5407.

[10] Abdelmoumin G, Rawat D B, Rahman A. On the performance of machine learning models for anomaly-based intelligent intrusion detection systems for the internet of things. IEEE Internet of Things Journal, 2021, 9(6): 4280-4290.

[11] Xu H, Sun Z, Cao Y. A data-driven approach for intrusion and anomaly detection using automated machine learning for the Internet of Things. Soft Computing, 2023, 27(19): 14469-14481.

[12] Alsheikh M A, Hoang D T, Niyato D. Optimal pricing of Internet of Things: A machine learning approach. IEEE Journal on Selected Areas in Communications, 2020, 38(4): 669-684.

[13] Woźniak M, Siłka J, Wieczorek M. Recurrent neural network model for IoT and networking malware threat detection. IEEE Transactions on Industrial Informatics, 2020, 17(8): 5583-5594.

[14] Zhao R, Gui G, Xue Z. A novel intrusion detection method based on lightweight neural network for internet of things. IEEE Internet of Things Journal, 2021, 9(12): 9960-9972.

[15] Sakar C O, Polat S O, Katircioglu M. Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks. Neural Computing and Applications, 2019, 31(10): 6893-6908.

[16] Liu Z, Liu Q, Xu W. Deep learning-based human motion prediction considering context awareness for human-robot collaboration in manufacturing. procedia cirp, 2019, 1(83): 272-278.

[17] Huang X, Sun J, Sun J. A car-following model considering asymmetric driving behavior based on long short-term memory neural networks. Transportation research part C: emerging technologies, 2018, 1(95): 346-362.

[18] Chen Y, Hu C, Wang J. Human-centered trajectory tracking control for autonomous vehicles with driver cut-in behavior prediction. IEEE Transactions on Vehicular Technology, 2019, 68(9): 8461-8471.

[19] Wijnands J S, Thompson J, Aschwanden G D P A. Identifying behavioural change among drivers using Long Short-Term Memory recurrent neural networks. Transportation research part F: traffic psychology and behaviour, 2018, 1(53): 34-49.

[20] Pei Z, Qi X, Zhang Y. Human trajectory prediction in crowded scene using social-affinity long short-term memory. Pattern Recognition, 2019, 1(93): 273-282.

[21] Chander N, Upendra Kumar M. Metaheuristic feature selection with deep learning enabled cascaded recurrent neural network for anomaly detection in Industrial Internet of Things environment. Cluster Computing, 2023, 26(3): 1801-1819.

[22] Belhadi A, Djenouri Y, Djenouri D. Group intrusion detection in the Internet of Things using a hybrid recurrent neural network. Cluster Computing, 2023, 26(2): 1147-1158.

[23] Thota S, Menaka D. Botnet detection in the internet-of-things networks using convolutional neural network with pelican optimization algorithm. Automatika, 2024, 65(1): 250-260.

[24] Latif S, e Huma Z, Jamal S S. Intrusion detection framework for the internet of things using a dense random neural network. IEEE Transactions on Industrial Informatics, 2021, 18(9): 6435-6444.

[25] Wu Y, Dai H N, Tang H. Graph neural networks for anomaly detection in industrial internet of things. IEEE Internet of Things Journal, 2021, 9(12): 9214-9231.